SVANTEK

USER MANUAL

SD 310

MONITORING SYSTEM CONTROLLER

Warsaw, 2022-08-04

Rev.1.01

Note: *Because of continuous product improvement SVANTEK reserves the right to make changes to product specifications without notice. To download the most up to date user's manual please visit our website at www.svantek.com. This user's manual presents the software revision* ***2.01.x****.*

The succeeding software revisions (marked with the higher numbers) can change the view of some displays presented in the text of the manual.

WEEE Notice: *Do not throw the device away with the unsorted municipal waste at the end of its life. Instead, hand it in at an official collection point for recycling. By doing this you will help to preserve the environment.*

Technical Support Contact Information:

web: www.svantek.com

e-mail: support@svantek.com.pl

CONTENTS

1 INTRODUCTION

SD 310 is an intelligent controller that allows the integration of a dust and weather measurement modules with Svantek monitoring systems. The unit enables physical connection of external devices such as dust meters or weather monitoring stations to provide them with a power and user interface.

SD 310 technology is based on a Raspberry Pi module, which is responsible for measurement control, saving measurement data from external devices in the form of data files on an SD card and for wireless external communication. Using the 4G, WLAN or LAN connection, SD 310 provides a user interface generated in the form of a website accessible via popular Internet browsers. The interface allows you to control the measurement (start/stop), configure measurement parameters, and view the measurement results.

SD 310 saves data from the peripheral devices as files that can be downloaded to a PC with the use of the *SvanNET* web service. The *SvanNET* web service gives also the full control of the system using any web browsing device like a PC, mobile phone or tablet.

SD 310 is dedicated to monitoring; therefore, it has been equipped with the ability to generate SMS and email alarms, based on current measurement data from the connected devices.

1.1 GENERAL FEATURES OF SD 310

- Provides power and interface to external monitoring devices such as dust meters or weather monitoring stations
- Creates files on a USB flash drive for easy use with the *SvanPC++* software
- Internal software enables easy configuration, measurement control, results preview and creation of measurement results files
- WLAN access point enabling wireless remote control via local network
- LTE (4G) modem enabling wireless remote control via *SvanNET* web service
- GPS module enables position log and time synchronization (optional)
- Internal AC/DC adapter for continuous power supply from 90-305 VAC
- Wide range of temperature operating conditions: from -20°C to +60°C
- Ingress Protection Rating – IP 65

1.2 ACCESSORIES INCLUDED

- Cellular modem
- 32 GB microSD card
- Cables according to the ordered configuration

Note: *Purchasing SD 310 entitles you to receive an access to the SvanNET Connectivity option, for which please contact your local distributor or SVANTEK sales department*.

1.3 ACCESSORIES AVAILABLE

- SV_USB32GB USB memory 32 GB
- SV_USB64GB USB memory 64 GB

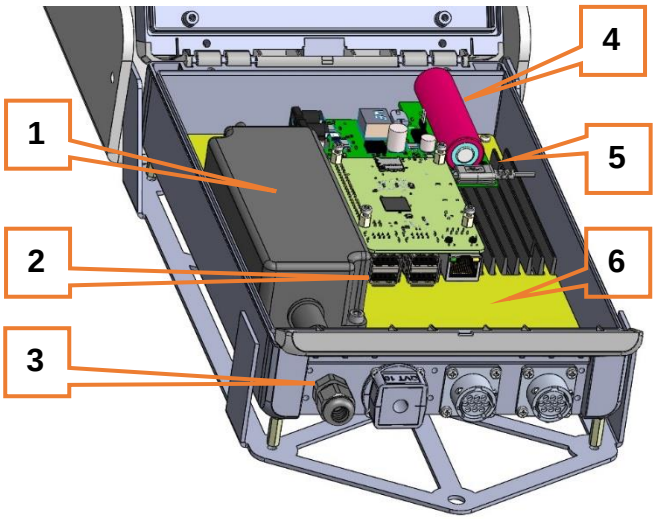
1.4 SUPPORTED DEVICES

- **SV 307A** noise monitoring terminal
- **SP 280** remote dust monitor
- **SP 276** weather station based on GILL module
- **SP 275** weather station based on Vaisala module

2 SD 310 HARDWARE COMPONENTS

The waterproof case of the SD 310 includes following components:

- Power supply module with AC/DC converter (1),
- Raspberry Pi single-board computer with four USB connectors and one LAN connector (2),
- Connectors (3),
- System UPS (4),
- Start/stop key and two LEDs (5),
- Space for the GSM modem and memory stick (6).



2.1 SINGLE-BOARD COMPUTER

The built-in single board computer has the following parameters:

- Quad Core 1.2 GHz 64-bit CPU
- 1 GB RAM
- WLAN - 2.4 GHz
- IEEE 802.11.b/g/n
- Bluetooth® - BLE 4.2[1]
- Cellular modem - LTE (4G)
- LAN - Gigabit Ethernet
- 4 x USB 2.0 ports
- full size HDMI
- 32 GB microSD card
- Linux OS

The USB ports can be used for connection the USB memory stick.

Optionally up to 3 x RS232 ports can be installed for the supported devices, like: dust monitor and weather station.

2.2 POWER SUPPLY MODULE

Built-in AC power supply unit capable to power of up to two external devices with input voltage 90 ÷ 305 VAC and output characteristics: 40 W, +15 V DC, 2.67 A.

Optional DC source: voltage range 10.5 V ÷ 17 V DC.

[1] "The *Bluetooth*® word mark and logos are registered trademarks owned by Bluetooth SIG, Inc. and any use of such marks by SVANTEK is under license. Other trademarks and trade names are those of their respective owners.

2.3 CONNECTORS

The connector panel has four connectors, which can be used in different configurations, for example:

1. 3-pin connector for the external power AC mains,
2. Cable glands for USB, LAN ect. connections,
3. Cable glands for USB, LAN ect. connections,
4. 8-pin External Interface port, incl. RS232 interface for the supported device (incl. power output 15 V, 1 A).

Note: *Switch the power off before connecting the unit to any other device.*

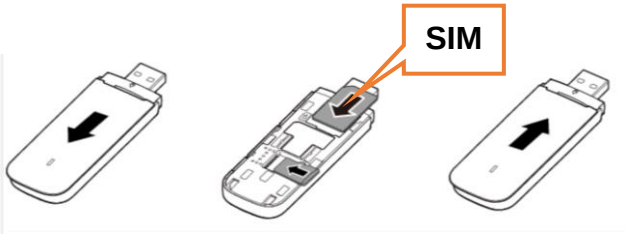
2.4 SIM CARD INSTALLATION

The Huawei modem is a USB Stick type LTE modem that is connected to the USB port with a dedicated cable.

To insert a SIM card, you need to open the SD 310 case, remove the modem from the slot and disconnect the modem from the USB cable.

Then slide the cover to reveal the place for the card. After inserting the SIM card, you need to put the cover back and slide it towards the USB connector.

After inserting the SIM card connect the modem to the USB cable, put it to the SD 310 case and close the case cover.



2.5 START/STOP KEY AND LEDS

Inside the SD 310 case there is a Start/stop key and two LED indicators, which show the status of power and the single board computer. Both key and LEDs are dedicated generally for the service staff.

Start/stop key

The Start/stop key inside the SD 310 case is used for turning on and off the unit when it is powered. One of the functions of this key is to hard reset the system when you press and hold the Start/stop for more than 10 seconds. This key is dedicated for the service stuff because the SD 310 is turned on or off by connecting it to the power source.

Note: *When the SD 310 is turning on, the Status LED indicator flashes orange several times. Do not press the Start/stop key, because the system will enter the service mode. In the service mode, the Status LED indicator flashes orange. If you entered the service mode, you could exit it pressing the Start/Stop key.*

Status LED

The Status LED is amber when the SD 310 is powered from the AC or DC power supply and the UPS battery is charging, green when the SD 310 is powered from the AC or DC power supply and the battery is not changing.

Power LED

The Power LED is white during the system start-up which lasts ca. 20 seconds. When the system has been started this LED turns green. When the system is closing it turns red for ca. 10 seconds and then goes out.

Note: *Basic turning the station on/off takes place without opening the box!*

3 INSTALLING MEASUREMENT SYSTEM

To install the measurement system:

1. Insert the SIM card
2. Connect the peripheral device(s)
3. Assure the 90-305 VAC or external DC power supply.

The SD 310 has a metal frame with holes for several screws to be installed at the wall in the position "connector panel on the bottom"

4 WORKING WITH SD 310

4.1 SYSTEM START-UP AND REMOTE CONTROL

Before connecting the SD 310 to the power supply, install the system – see Chapter 0. After connecting the power supply, the SD 310 will turn on automatically and simultaneously turn on all peripheral devices.

The SD 310 carries out a self-test and connects to the *SvanNET* server if an Internet connection was proper configured. Otherwise, you must configure the Internet connection using a WLAN connection.

Basic operations for controlling the system and setting up peripheral devices include:

- Measurements start/stop
- Measurement results viewing
- System check (memory free space, battery, wireless connection etc.)

SVANTEK offers two tools to support the SD 310 functions:

- Internal SD 310 software for Internet connection configuration, measurement parameter settings, measurement results preview, measurement start / stop and system status monitoring
- SvanNET web service for system status monitoring and access to the SD 310 device.

Apart from the control operations of the SD 310 measurement system, SVANTEK offers the *SvanPC++* software for processing files and data generated by the SD 310.

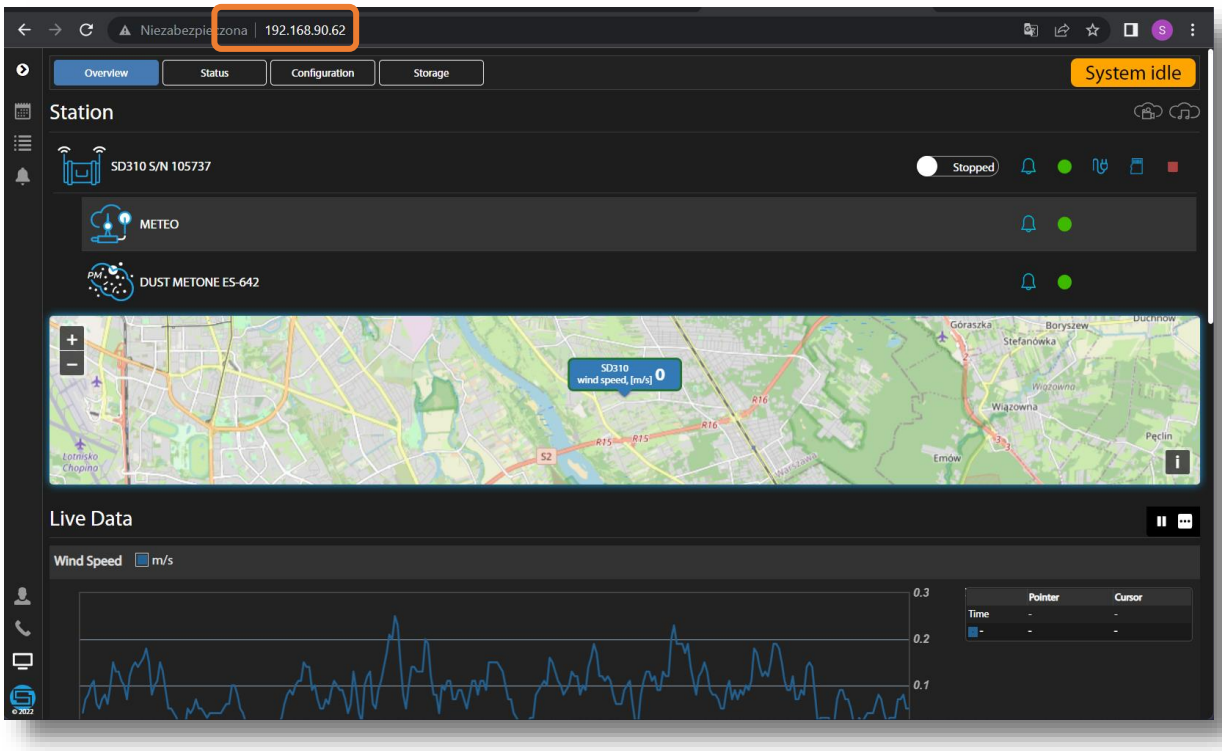
4.2 SD 310 INTERNAL SOFTWARE

The internal software of the SD 310 is dedicated for configuring the measurement system and connection with the Internet, previewing results and creating files with results. The SD 310 has its own website providing the user with access via a local WLAN network or via the *SvanNET* web service, which can redirect traffic to the server's website.

The user can enter the SD 310 website on a mobile device connected to the server's WLAN network via a web browser by entering the text "svantek.local" or the IP address “192.168.90.” in the WWW field.

After opening the WWW website of the SD 310, you are getting access to the control functions which enable you to overview the results (**Overview**), check the instrument status (**Status**), configure instrument settings (**Configuration**) and manage data storage (**Storage**).

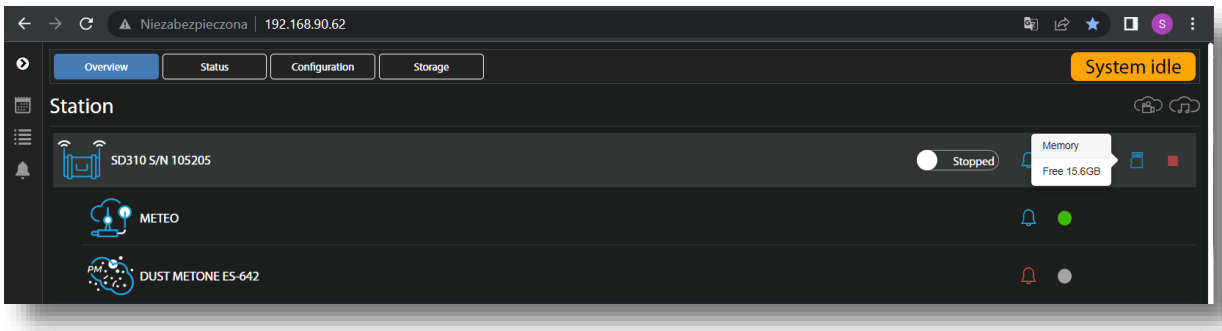
Note: *The screens presented below are made for SD 310 with two RS232 connectors allowing to use two monitoring stations – Dust and Meteo.*



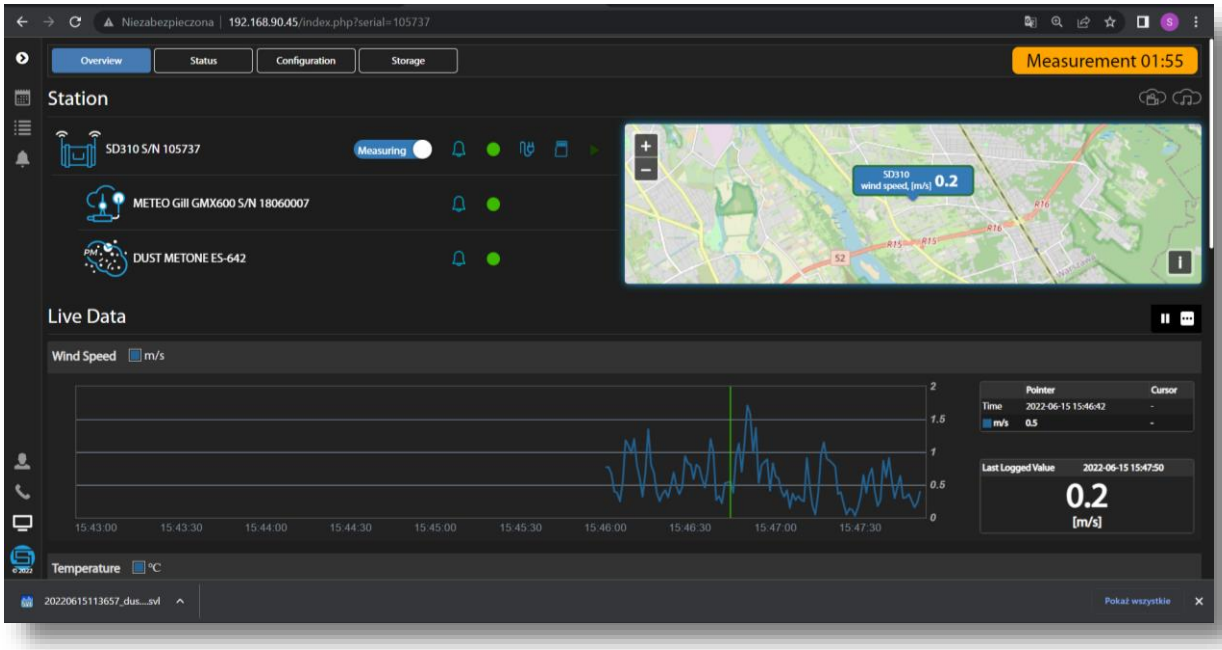
4.2.1 Results Overview

The **Overview** window consists of two sections: **Station** section presenting configuration of your station together with the map showing its location and Live Data section presenting time-histories of measured results.

In the **Station** section, you can start/stop the measurement and check their status, clicking on the icon at the device bar. When the measurement is stopped, the “System idle” label appears at the upper right corner.

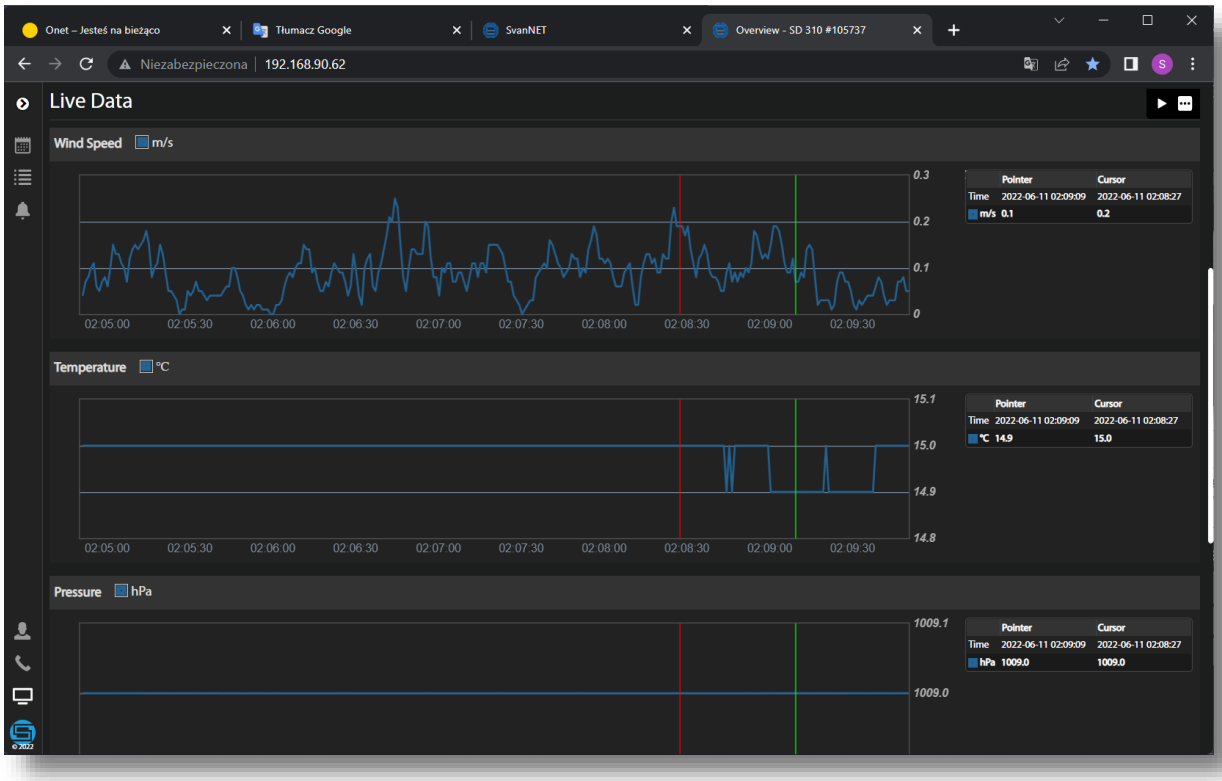


If the measurement is running, this label changes to the elapsed time. The Live data section starts to present measured results as time-history plots.

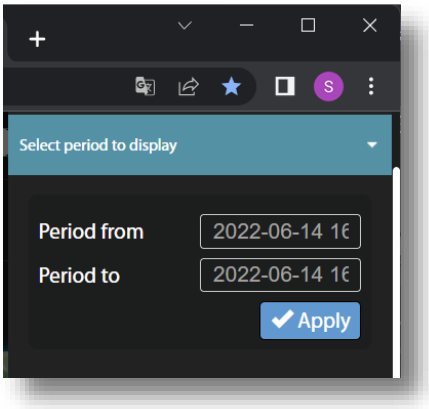


In the **Live Data** section, you can read-out data values using the pointer and the cursor. To set the cursor, point your mouse and left-click.

To see more time-histories, scroll the window.



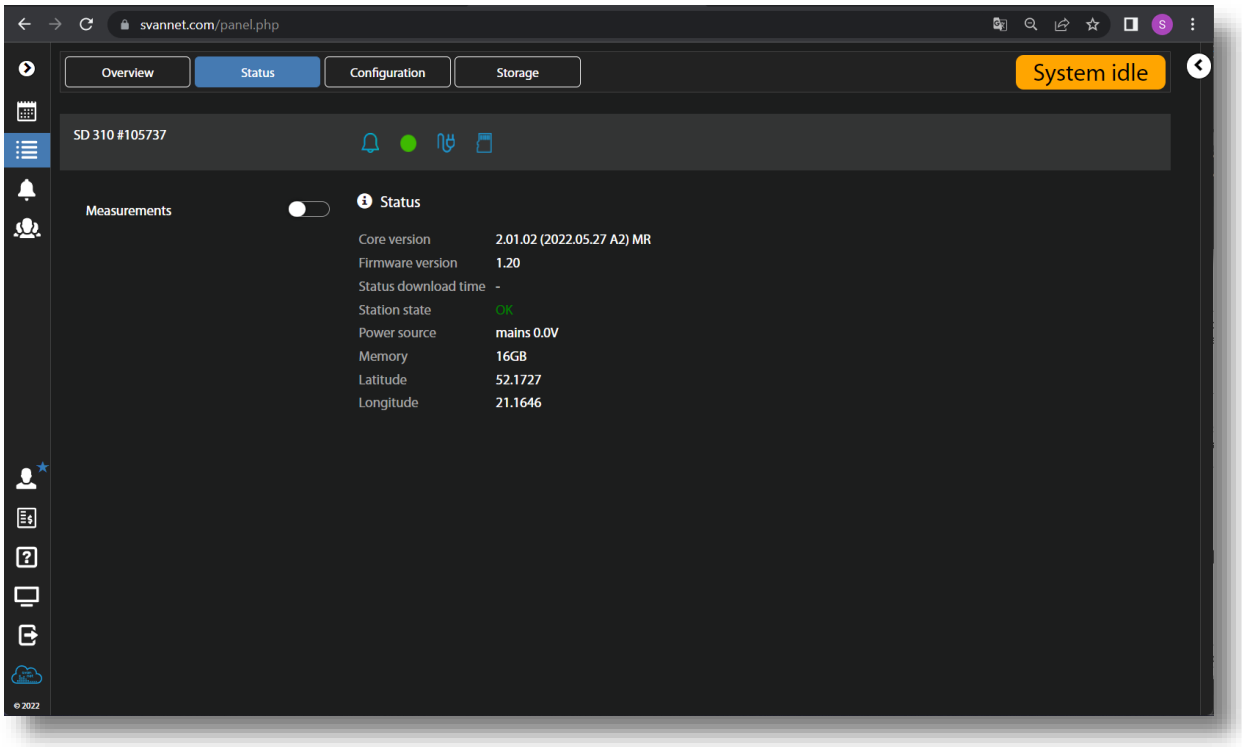
Click the  icon to choose the period for data presentation.



4.2.2 Checking Status

The **Status** window shows the status of SD 310, meanly: version of the core processor, firmware version, status download time, state of the station, power source, memory and GPS co-ordinates.

In the **Status** window, you can start/stop measurements.

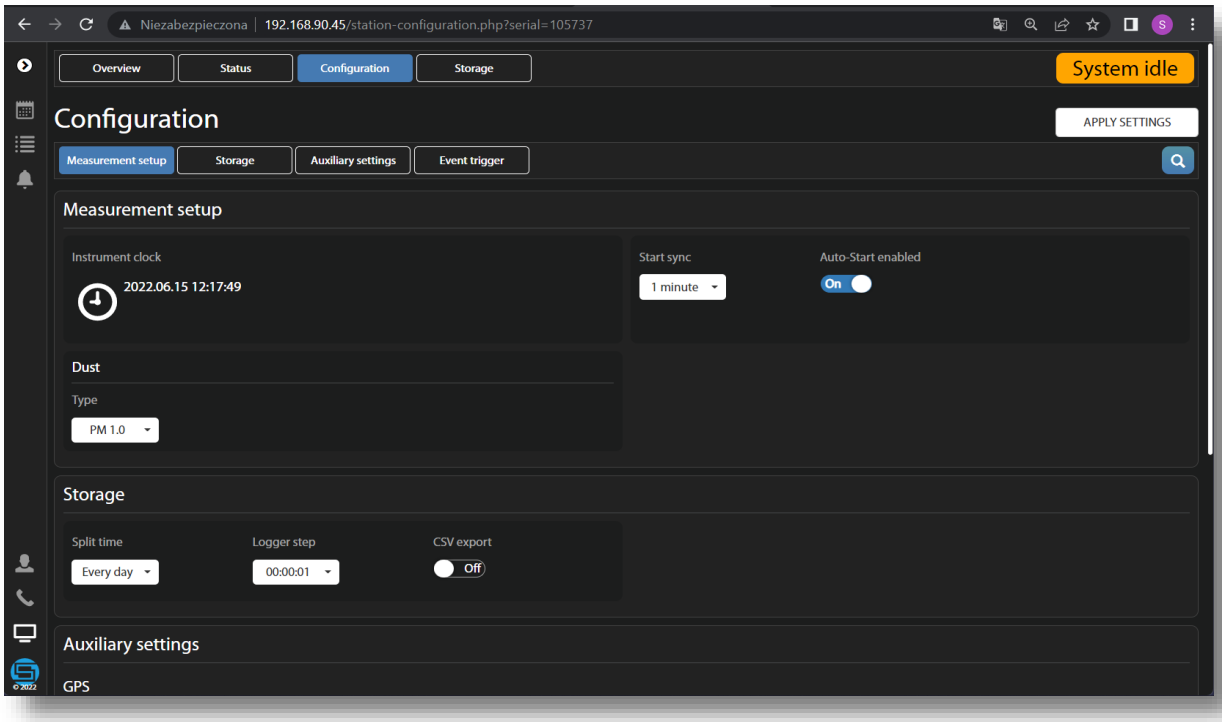


4.2.3 Configuring the station

The **Configuration** window allows you to configure the measurement and station settings. It includes four sections: **Measurement Setup**, **Storage**, **Auxiliary settings** and **Event trigger**.

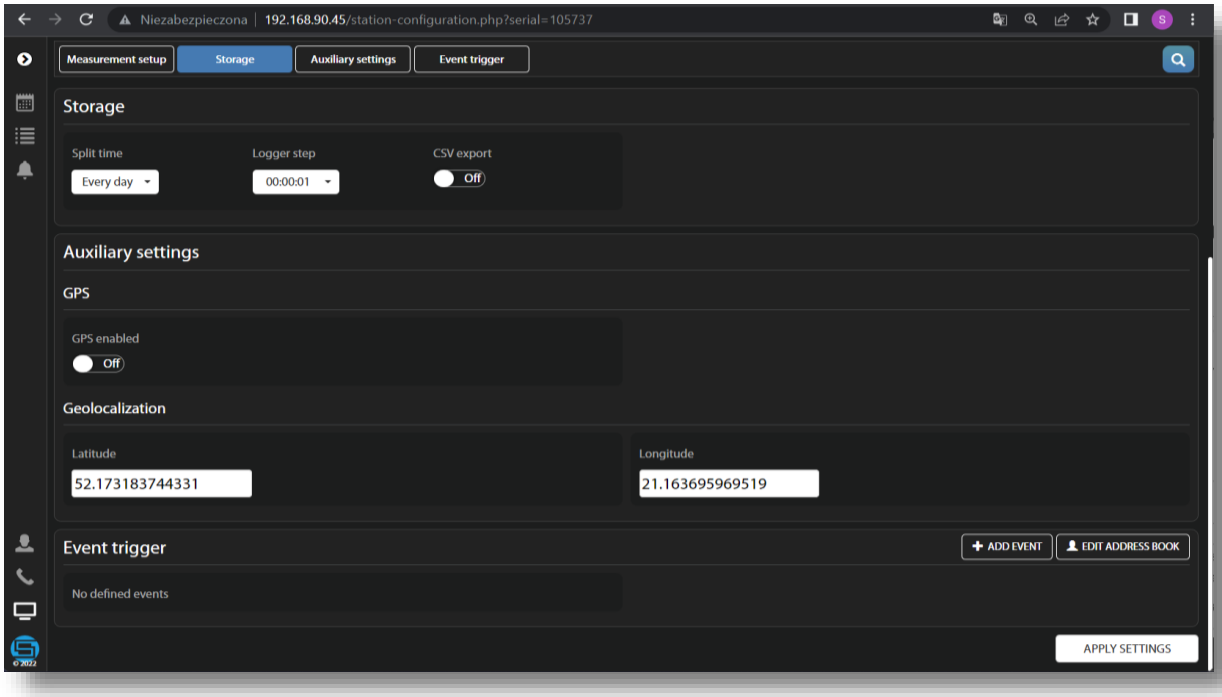
In the **Measurement setup** section, you can:

- Set synchronization of the measurement start (**Start sync**): *Off*, *1 second*, *1 minute*, *15 minutes* or *1hour*.
- Enable/disable the automatic start when the instrument is turned on (**Auto-Start enabled**).
- Select **Type** of the **Dust** sensor: *PM 1.0*, *PM 2.5*, *PM 10.0* or *TSP*.



In the **Storage** section, you can:

- Split data files (**Split time**): *Every 15 m*, *Every 30 m*, *Every 1h* or *Every day*.
- Select the **Logger step**: 00:00:01 ÷ 00:54:00.
- Switch On/Off export of data files in the CSV format (**CSV export**).



In the **Auxiliary settings** section, you can:

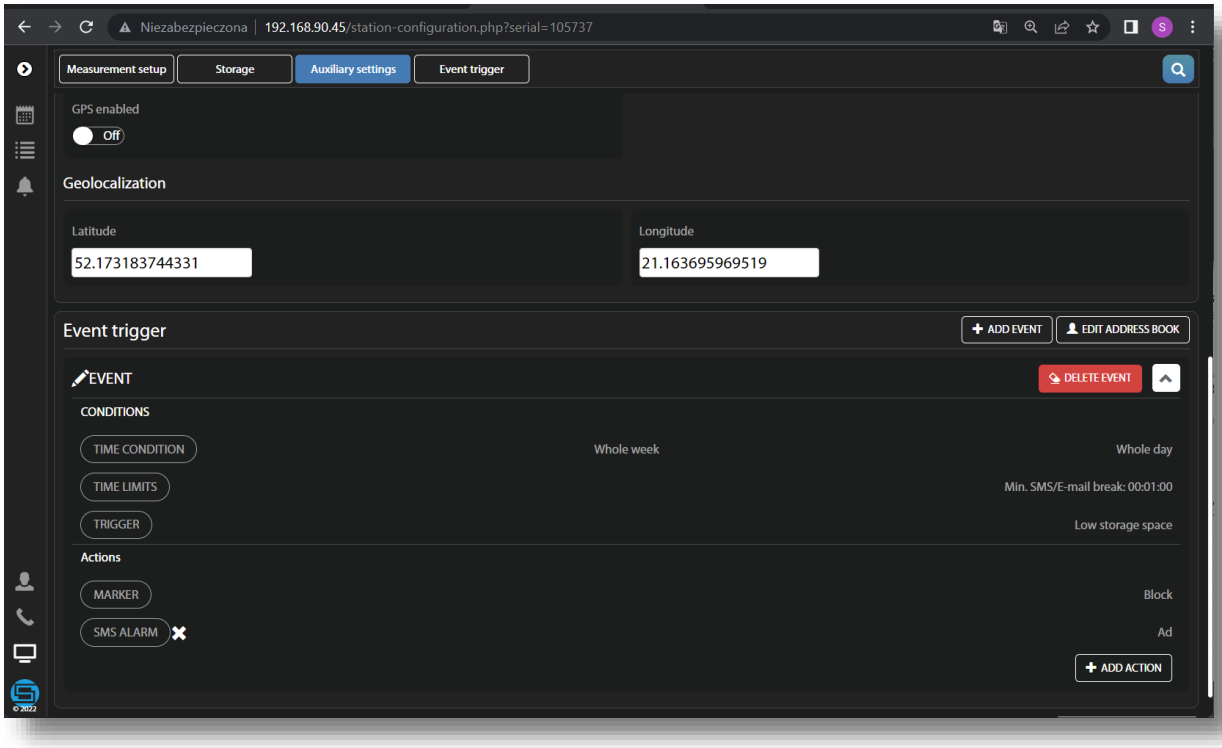
- Switch On/Off the **GPS**.
- For disabled GPS set the **Latitude** and **Longitude** manually.

In the **Event trigger** section, you can:

- Add new event for generation actions – markers or alarms (**+ADD EVENT**).
- Edit addresses of alarms recipients (**EDIT ADDRESS BOOK**).

4.2.4 Configuring events

After adding the event, additional section for this event appears allowing you to change the default event name and configure event **CONDITIONS** and **Actions**.



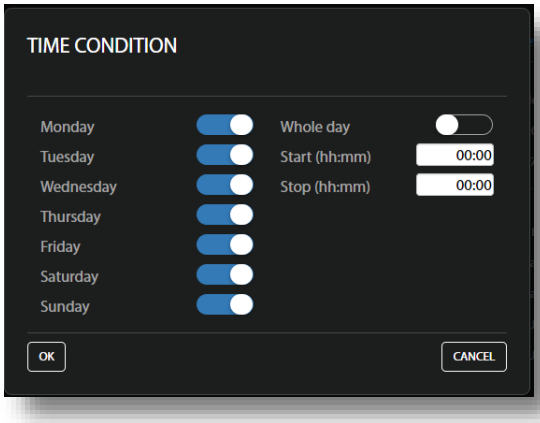
- Events are specified as a combination of superimposed **CONDITIONS** (logical AND) such as specific time intervals (**TIME CONDITION**) in which measurement threshold levels are exceed or system events occur in logical OR (**TRIGGER**) taking into consideration minimum period of break between SMS and E-mail notifications (**TIME LIMITS**).

- Each Event may be connected with special actions (**Actions**) such as: block marker recording to the logger file (**MARKER**), sending SMS with alarm notification (**SMS ALARM**) or sending e-mail with alarm notification (**E-MAIL ALARM**).
- There is an address book containing SMS and E-mail recipients addresses. You can edit this book clicking **EDIT ADDRESS BOOK**.

You can configure **CONDITIONS** and **Actions** using the appropriate buttons. Settings are presented in the button line.

For example, the EVENT configuration in the above screen means that the event will appear when the data storage space is low. The occurrence of such an event will generate alarms throughout the week and will be accompanied by marker recording and sending SMS with an alarm notification.

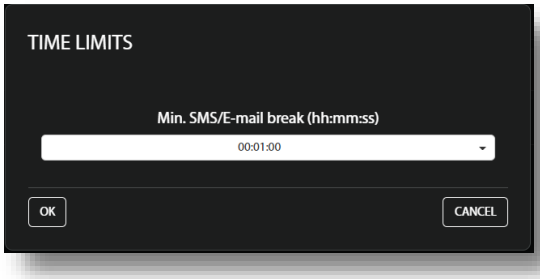
Configuring Conditions



If you click the **TIME CONDITION** button, the TIME CONDITION configuration box will pop-up.

In this box, you can select days and periods for events registration.

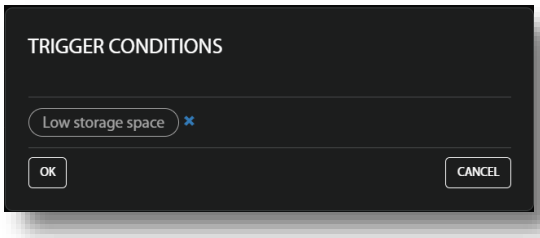
Press **OK** to confirm settings.



If you click the **TIME LIMITS** button, the TIME LIMITS configuration box will pop-up.

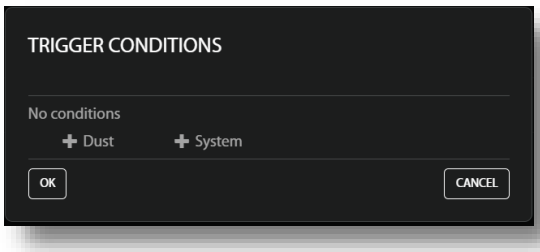
In this box, you can select minimum period of break between SMS and E-mail notifications. This enables to avoid hail of alarms in case of frequently recurring events.

Press **OK** to confirm settings.



If you click the **TRIGGER** button, the TRIGGER CONDITIONS configuration box will pop-up.

If you already have selected condition it will appear in this box.



If you want to change the condition, you should cancel current condition and select the new one. this box, you can add the condition type: **Dust** or **System**.

These conditions are mutually exclusive for the same event.

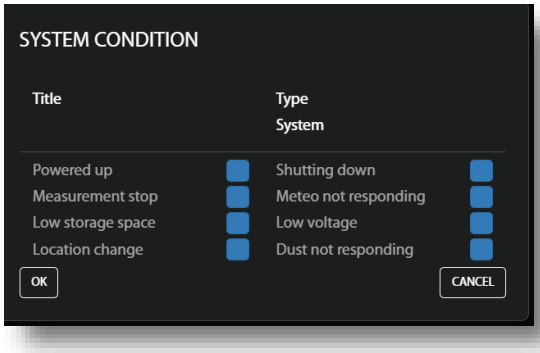
Dust trigger condition

The **Dust** type trigger activates the event when some result (**Source**) exceeds the **Threshold** level.

System trigger condition

The **System** type trigger activates the event when some of the system conditions appear.

You can select several or all conditions.



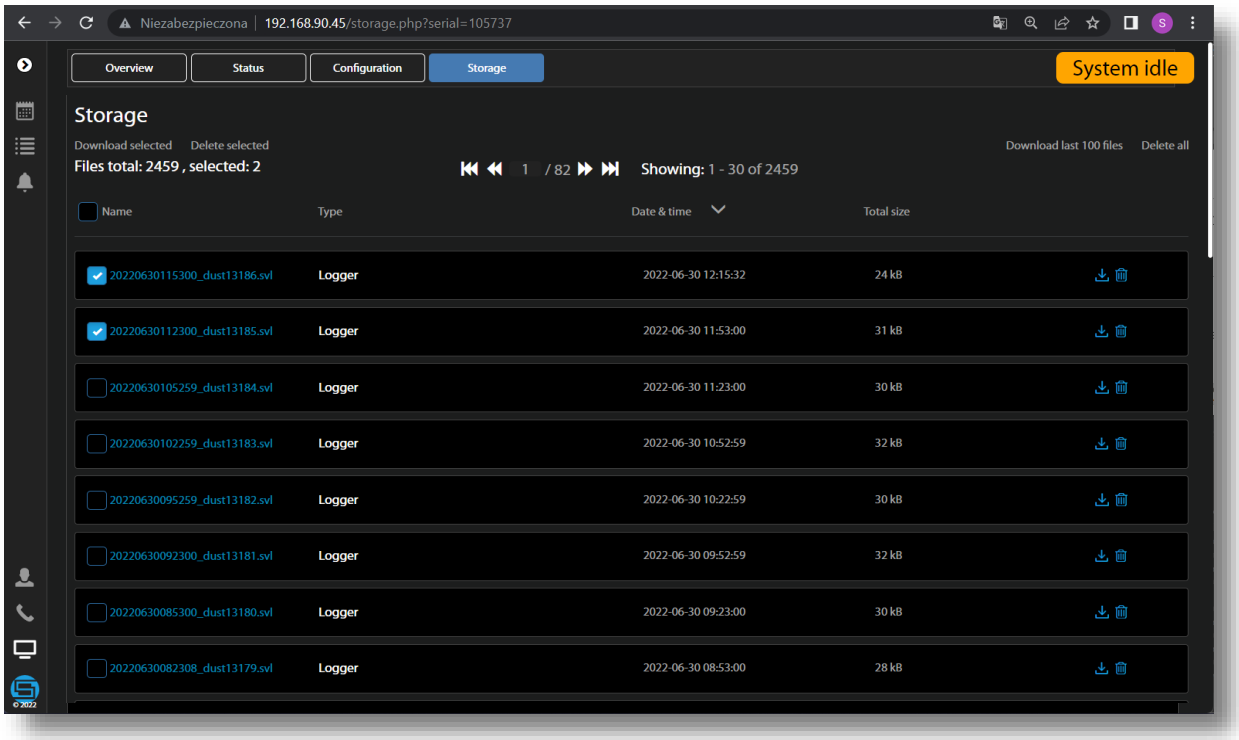
The SYSTEM CONDITION alarms are generated when:

- the system is turned on (**Powered up**)
- the measurement is not running for any reason (**Measurement stop**)
- free memory drops below 20% (**Low storage space**)
- the GPS indication is changed **(Location change**)
- SD310 is turned off (**Shutting down**)
- the Weather station is not responding for 2 minutes (**Meteo not responding**)
- the battery capacity drops below 10% (**Low voltage**)
- the Dust monitor is not responding for 2 minutes (**Dust not responding**)

Note: *After making all necessary settings, press the APPLY SETTINGS button.*

4.2.6 Managing files

The **Storage** window allows managing files stored in the SD 310 memory.



In the **Storage** window, you can:

- Download or delete individual files by clicking the righthand icons on the file line.
- Select several files and download or delete selected files.
- Download or delete all files.

Sort files in the list by **Name**, **Type**, **Date & time**, **Total size**.

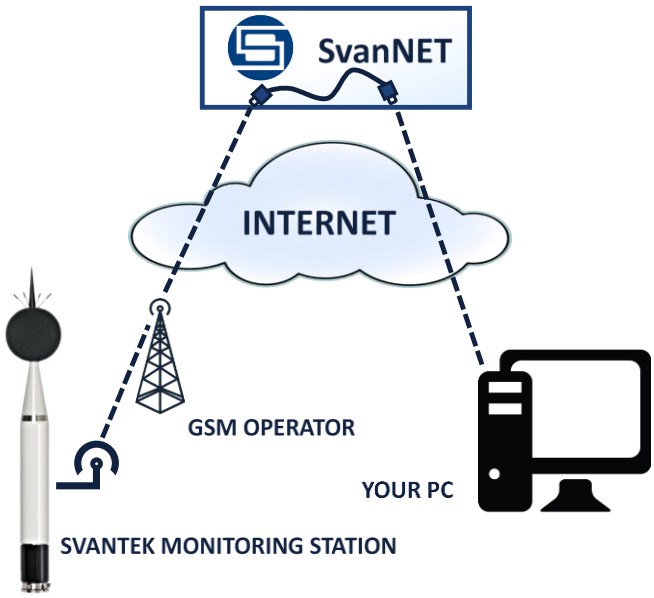
Files are downloaded to the directory on your PC according to the browser rules.

4.3 SVANNET WEB SERVICE

SvanNET is an Internet service that simplifies the remote connection between PC and Svantek devices and offers you simple access to the instrument's settings, results, and status information.

SvanNET allows usage of all type of SIM cards with a GSM modem regardless of having a public or private IP.

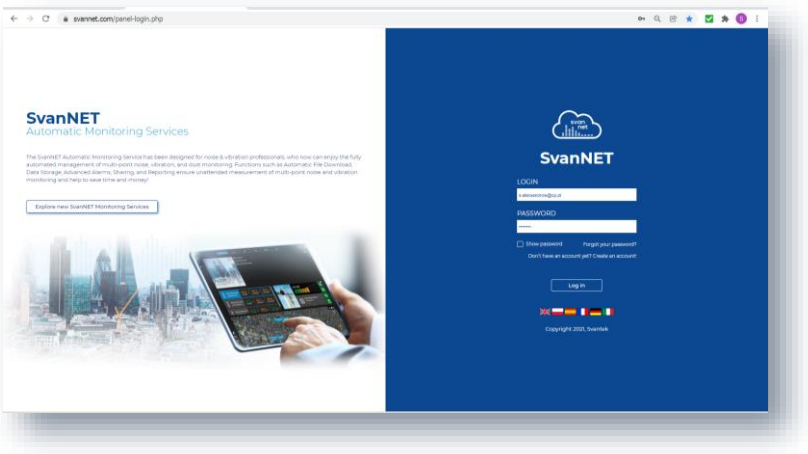
Note: *Establishing GSM connection requires usage of a SIM card with no PIN protection with activated Internet access.*



Note: *To have access to the* ***SvanNET*** *web service the local SVANTEK distributer should create the user's account and assign monitoring stations to it.*

To access *SvanNET*, log in to your account at: https://www.svannet.com/panel-login.php

Before logging, select your language.



SvanNET includes the standard function - *Remote Communication Services* available for all the *SvanNET* users and the optional extension - *Automatic Monitoring Services* offered via a license.

Remote Communication Services maintain remote connection with the monitoring devices and service includes status alarms (e.g. battery, memory), remote access to device settings and measurement files stored in the device and preview of the current results and recent time-history graph.

Automatic Monitoring Services offers automatic control of measurement points, data sharing with other *SvanNET* users as well as data preview in the form of a customised website with either public or restricted access. The preview website can be customised with a logo and individual project name. Access to the preview can be either open to the public or protected by a password.

You can switch both services using icons on the Main panel:

– *Automatic Monitoring Services* (**Project list**)

– *Remote Communication Services* (**Station list**).

If you have extended *SvanNET* package, you can use both tools. If you have standard *SvanNET* package, only Station list tool will be available.

***Note**: This manual describes only the **Station list** tools. To get more information about **Project list** see SvanNET User Manual.*

4.3.1 Station list view

Station list displays all stations assigned to your account – turned on and off. When you click the station, it becomes active and the tools at the right panel will be dedicated to this particular station.

The station bar except station name with serial number includes six icons that indicate the station state. When a station is disconnected from *SvanNET* all icons are of grey colour.

If you click the station name, station information will be displayed. If you click the icon, this icon status information will be displayed:

Project status: this icon appears when this station is involved in the project. When you click this icon, the project name and link to it will be displayed.

Alert status: blue - everything is OK, red – unregular event is happening.

Station connection status: green – online; grey – offline; yellow - the station doesn't respond to the command for a long time.

Battery state. When you click this icon, the information about battery state will be displayed.

External power source status: blue – the instrument is powered by the external source, grey - there is no external power.

Memory status. When you click this icon, the information about available memory will be displayed.

Icons in the Main panel tool allows you to:

display alarms for all stations

manage user account

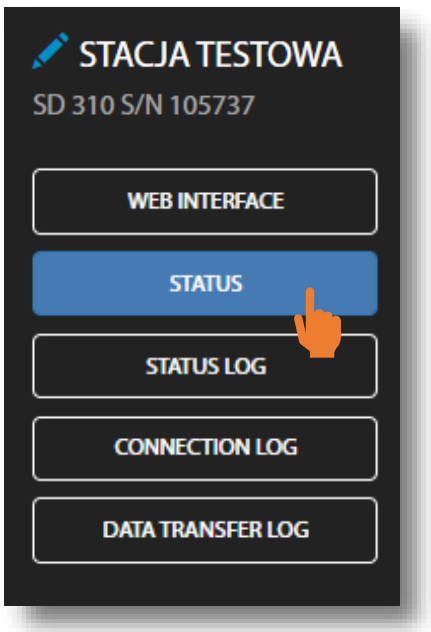
activate licences

contact Svantek Support team

change the colour scheme of *SvanNET* from "dark" to "light"

logout from *SvanNET*.

The Tool panel provides some functions for station control. To switch the function, point a cursor on the appropriate button (it will change its colour to blue) and click it.



The **WEB INTERFACE** switches to the internal program WWW of the SD 310 (see Chapter 4.2) which allows you to view the recorded results, configure and start/stop measurements. This button is available for stations connected to *SvanNET*.

The **STATUS** button switches you to the Station status view (see Chapter 4.3.2) in which you can check the station status and configure status alarms.

The **STATUS LOG** button switches you to the Status log view (see Chapter 4.3.3) in which you can check the power source (type and charge level), memory free space and GSM signal quality.

The **CONNECTION LOG** button switches you to the Connection log view (see Chapter 4.3.3) in which you can check the history of station connections.

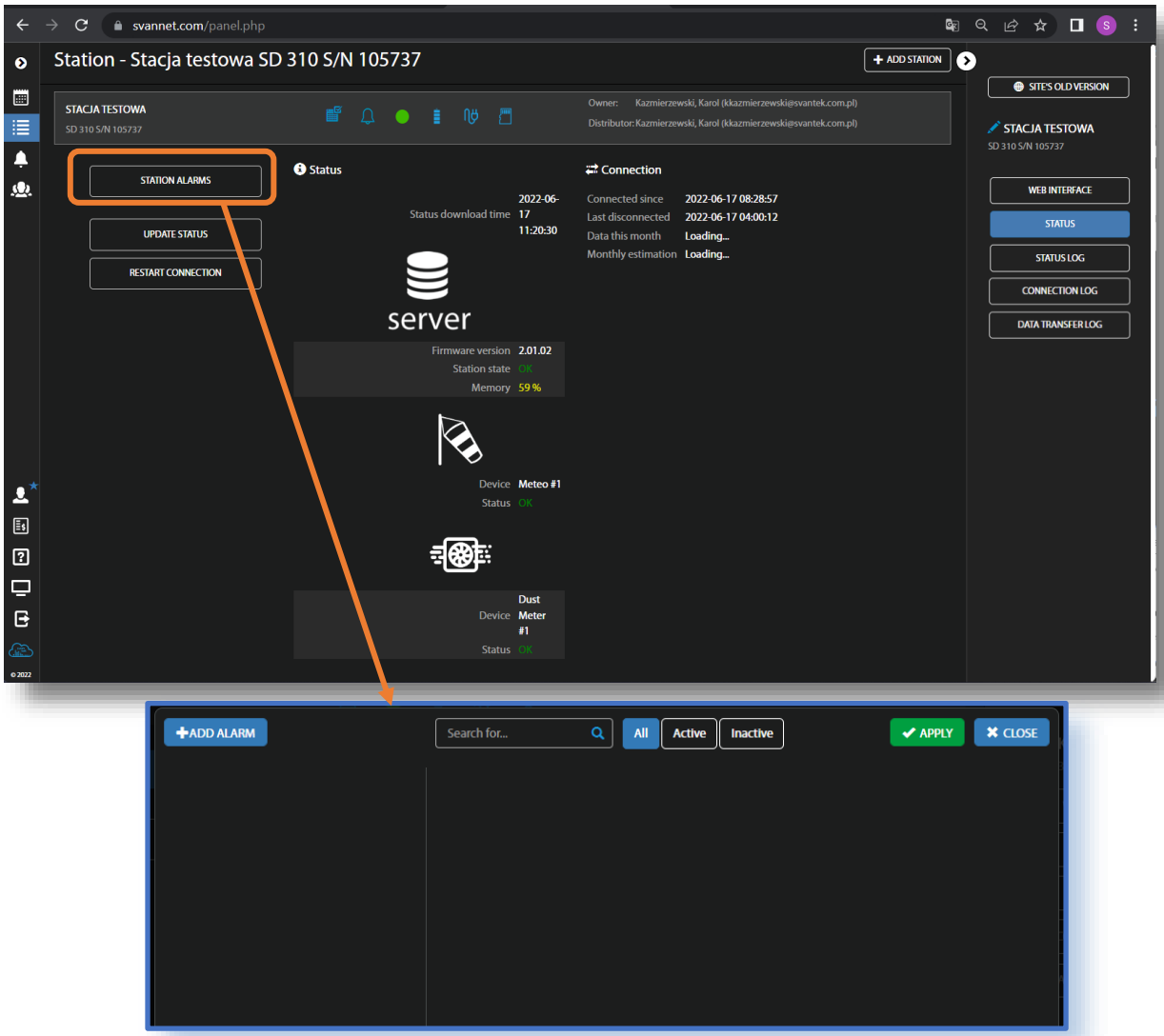
The **DATA TRANSFER LOG** button switches you to the Data transfer log view (see Chapter 4.3.3) in which you can check the history of data transfers.

Clicking  you can set the new station name instead of the default.

4.3.2 STATUS view

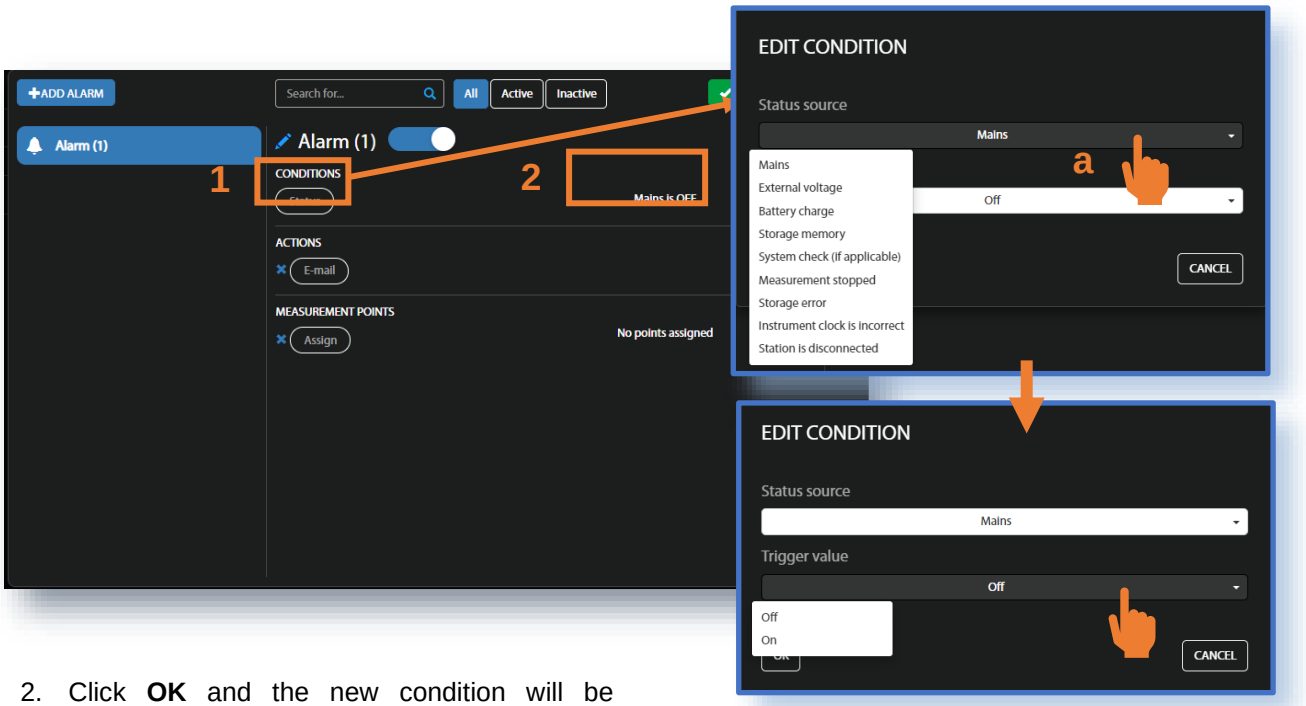
In the STATUS view, you can check the status of the SD 310 system components and configure alarms for the respective states that *SvanNET* will generate and send to the defined recipients in the form of e-mail.

- To update the system status, click the UPDATE STATUS button.
- To configure the condition of state alarms and the related actions, click the STATION ALARMS button.



After clicking STATION ALARMS, click **+ADD ALARM** in the pop-up box and a new **Alarm(1)** with CONDITIONS, ACTIONS and MEASUREMENT POINTS settings will appear. Alarms are based on Conditions and relate to Actions, that are default e-mails to the specified recipients, and refer to Measurement points. To configure Alarm:

1. Click the **Status** button and in the EDIT CONDITIONS configuration box:
   a. select **Status source**: **Mains**, **External voltage**, **Battery charge**, **Storage memory**, **System check** ect.,
   b. click the **Trigger value** selector and choose the required value of the selected **Status source**.



2. Click **OK** and the new condition will be displayed in the CONDITIONS area.

The *SvanNET* alarms have next meanings:

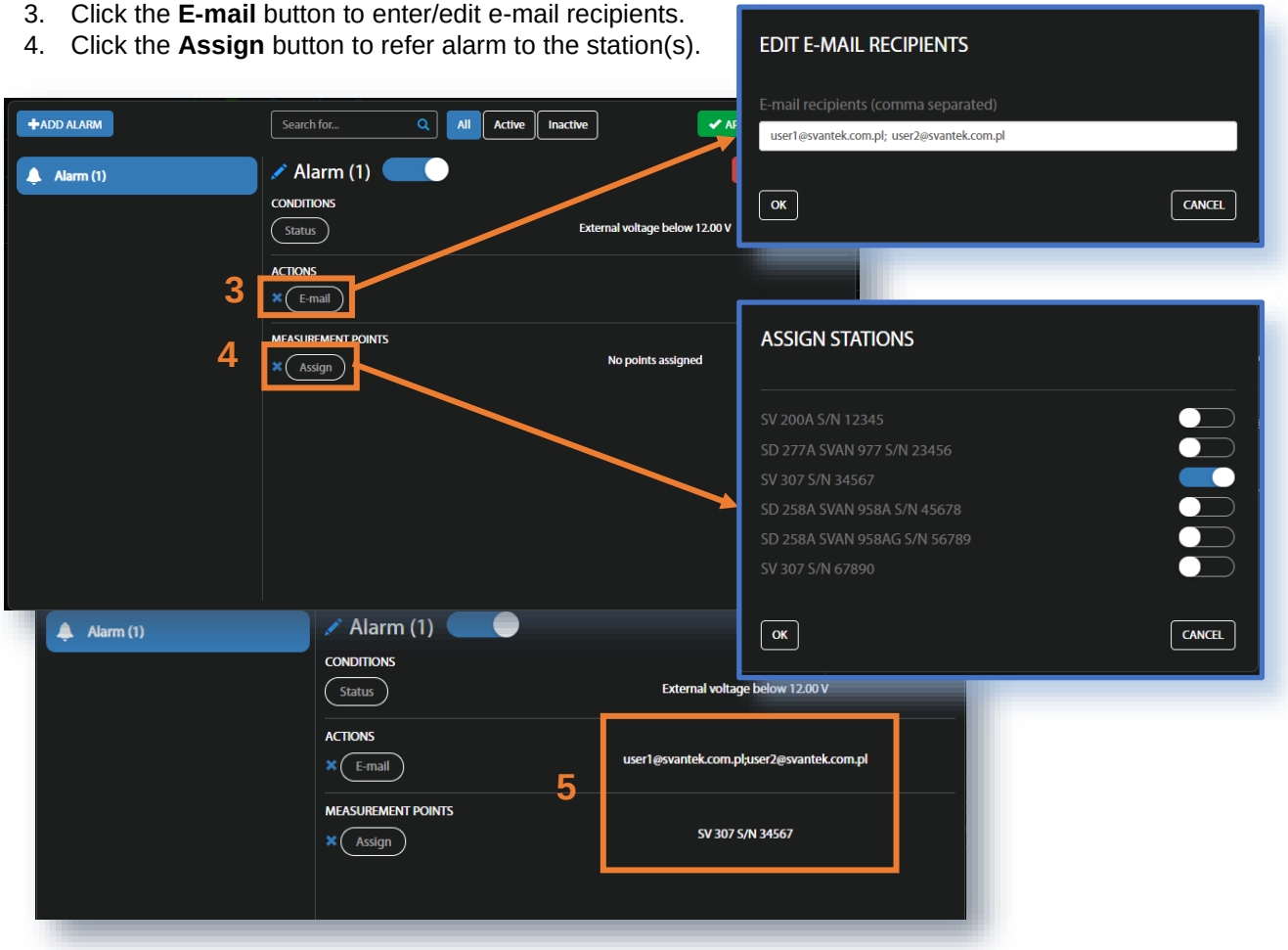
- **Mains**
  - Trigger Value: Off – alarm is generated when the system detects loss of power supply
  - Trigger Value: On – alarm is generated when the system detects appearance of power supply
- **External voltage**
  - Trigger Value: xx.xx V – alarm is generated when the system detects an external power drop below the selected value. In this case, external power means power supply and all various battery packs
- **Battery charge**
  - Trigger Value: xx % - alarm is generated when the system detects a decrease in the percentage of battery charge below the selected threshold.
- **Storage memory**
  - Trigger Value: xx MB/GB - alarm is generated when the system detects a decrease in the free storage memory below the selected threshold.
- **System check (if applicable)**
  - Alarm is generated when the system detects failure in execution of the system check procedure (not live check).
- **Measurement stopped**
  - Alarm is generated when the system detects lack of measurement. Applies only to stopped measurements - states such as start delay, waiting for synchronization and pause are treated as a running measurement
  - Instrument action: Start measurement
- **Storage error**
  - Alarm is generated when the system detects an SD-card error. The check assumes that a measurement is in progress and data are recorded; the writing of the logger file is checked by changing of the free space on the card (which means that the device is writing data).
  - Instrument action: Restart measurement
- **Instrument clock is incorrect**
  - Trigger value: xx seconds / xx minutes – alarm is generated if the RTC indication of the device is inconsistent with the current system time (based on owner's time zone) by ± of the selected value

- Instrument action: Set instrument clock to server time (based on owner's time zone) – measurement is stopped, instrument clock is set (based on owner's time zone), measurement is resumed

- **Station is disconnected**
  - Trigger value: xx minutes / xx hours – alarm is generated when the station remains disconnected from *SvanNET* for a time equal to the selected value.

Alarms are reported once after the occurrence of an alarm condition. The occurrence of an alarm condition will generate selected actions (e.g. e-mail) at the moment of changing the status compared to the previous check (i.e. if at 8:15 there is power supply, at 8:30 mains is off, at 8:45 mains is still off, the system will generate an alarm at 8:30 and will be still until mains is on and off again).

3. Click the **E-mail** button to enter/edit e-mail recipients.
4. Click the **Assign** button to refer alarm to the station(s).



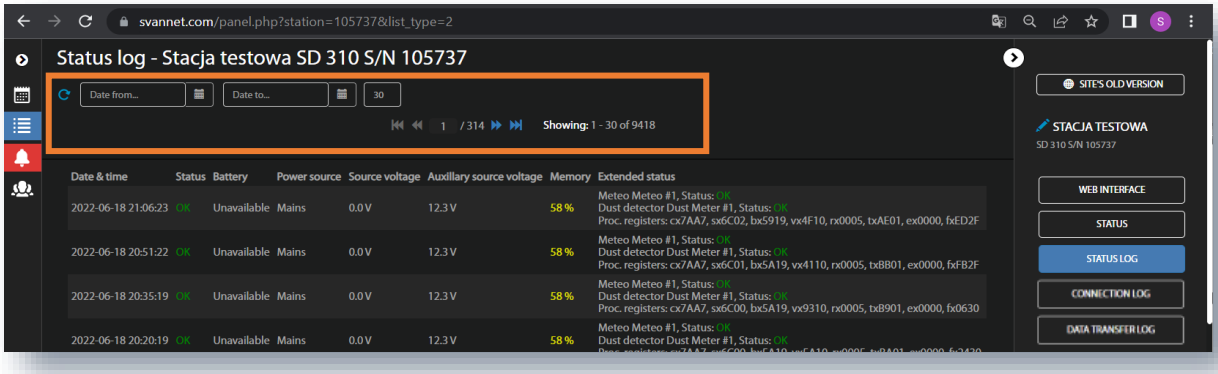
5. Made selections are displayed in the ACTIONS and MEASUREMENT POINTS sections.

4.3.3 LOG views

There are three station logs, that register system events, connections and data transfer:

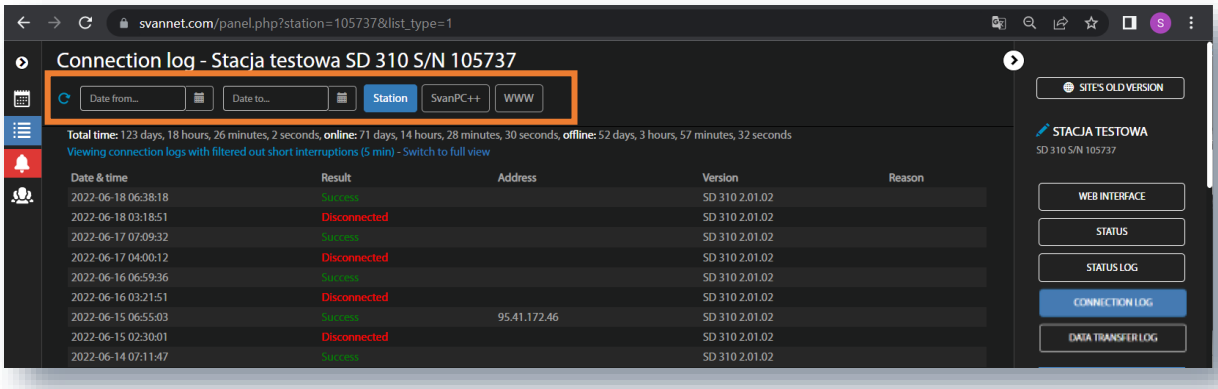
- **Status log** which registers power source (type and charge level), memory free space, GSM signal quality, system check history and GPS information.

  In the upper line, you can: refresh the log, select the required period of records to be displayed and rewind records.



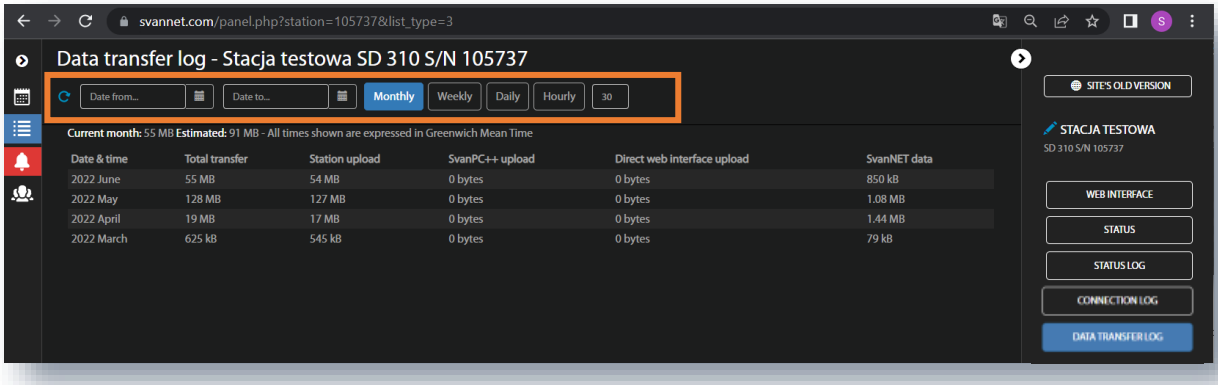
- **Connection log** which registers history of station connections.

  In the upper line, you can: refresh the log, select the required period of records to be displayed and rewind records.



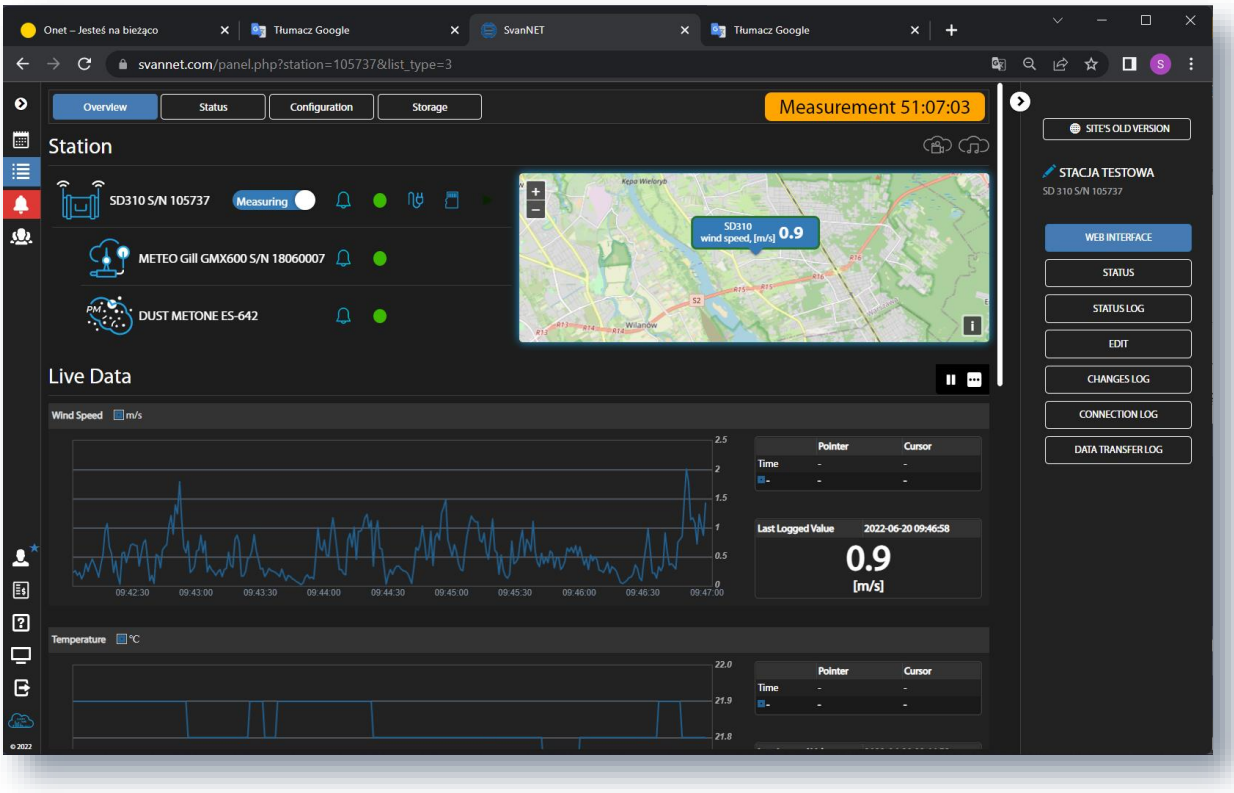
- **Data transfer log** which registers history of data transfers (uploads).

  In the upper line, you can: refresh the log, select the required period of records to be displayed and select the period for data transfer presentation: Monthly, Weekly, Daily or Hourly.



4.3.4 WEB INTERFACE view

The WEB INTERFACE button switches on the functionality described in Chapter 4.2, which enables viewing measurement results, configuration and starting/stopping measurements.



4.4 ANALYSING RESULTS USING SVANPC++

The SD 310 saves files from peripheral devices (monitoring stations - .svl and .wav, video cameras - .mp4, radar - .svl) in the project directory on the storage device (e.g. USB flash drive). Each project directory contains an additional file with the name projXXX.spr, containing information about the files of this project. Additionally, in the root directory on the storage device, the SD 310 saves a special file called "start.spr" which contains data about all projects on this storage device.

To open a project in the SvanPC++ program, connect the storage device to the PC, copy the project directory to the PC if necessary and double-click the projXXX.spr or start.spr file.

Note: *See SvanPC++ User Manual.*

Appendix A. SD310 API

A.1. BASIC INFORMATION

SD310 is a TCP-based server. You can build your own TCP client application or service which can connect to and converse with the SD 310 server.

Your application can send commands and receive responses from the SD 310 server.

By using these commands, you can:

- check the server status
- check the status of connected measuring devices
- create a new measurement project
- set the measurement parameters
- configure the system according to the project parameters
- start the measurement
- monitor the system status during the measurement
- configure alarms to be sent as SMS or E-mail
- read current measurement data
- stop the measurement.

Below you will find a description of the API requests and an examples program code in C# using which you can connect to the SD 310 server by passing the address and port of the server and then send a command and receive the response from the server.

A.2. STRUCTURE OF TRANSMITTED DATA

Server commands and responses are sent in JSON format.

For example:

```
{
  "Type" : "C1",
  "Request" : "Projects-New",
  "Parameters" : {
    "uid" : "Test 1",
    "descr" : "descr. of the project ..."
  }
}
```

JSON object literals are surrounded by curly braces {}.

JSON object literals contain key/value pairs.

Keys and values are separated by a colon.

Keys must be strings, and values must be a valid JSON data type:

- string
- number
- object
- array
- boolean
- null

Each key/value pair is separated by a comma.

A.3. STRUCTURE OF REQUESTS / COMMANDS

Queries / commands are JSON structured:

Key	Value
Type	“C1”
Request	one of the command list, e.g."`Projects-New`"
Parameters	JSON, each command has its own parameters
Token	token - string obtained from the server during login

For example:

```
{
  "Type" : "C1",
  "Request" : "Projects-New",
  "Parameters" : {
    "uid" : "Test 1",
    "descr" : "descr. of the project ..."
  }
}
```

A.4. STRUCTURE OF RESPONSES

Responses are JSON structured:

Key	Value
Status	command execution status, possible values “OK”,”ERROR”, “WARNING”
StatusMsg	comment or error description if the status is different from "OK"
Data	data returned by the server in JSON ARRAY format

For example, in response to the command "Projects-New" we will get a reply:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "Data": [
    {
      "id": 1073,
      "uid": "Test 1",
      "status": "",
      "time_created": "2021-06-17 05:56:00",
      "time_modified": "2021-06-17 05:55:59",
      "venue_gps_lat": "52.172703",
      "venue_gps_lon": "21.164102",
      "dev_settings_2": {},
      "alarms": {}
    }
  ],
  "Errors": []
}
```


A.5. COMMAND LIST

Request	Description
System-Login	login to the system
System-ChangePassword	allows you to change the password

System-GetStatus	returns basic information about the state of the system
Devices-GetList	returns a JSON array, information about the server and connected devices
Projects-GetList	returns a JSON array, returns a list of projects
Projects-New	creates a new project
Projects-Update	allows you to change project data, e.g. title, description, place of measurements
Projects-Cancel	sets flag "project to delete"
Projects-Restore	clears the "project to delete" flag
Projects-Activate	configures the system for measurements according to the project settings
Measurements-Start	starts the measurement
Measurements-GetTimeHistData	allows you to download the current measurement data from the given time range
Measurements-Stop	stops the measurement
? Alarm	
? Config - language	

A.6. TYPICAL SEQUENCE OF ACTIONS

An exemplary sequence of commands designed to perform the measurement can cause follows:

- **System-Login**
  Enter the login and password in the parameters and in response we get, for example, "Token": "112120938120", which we append to each subsequent command.
- **System-GetStatus**
  Allows you to check the general state of the system. In response, we get a set of information. We can start a new measurement if we get from the server: "status": "idle", "tech_alert_level": "ok".

- **Devices-GetList**
  Allows you to check in detail the status of the server and connected devices,
  important if the overall condition of the system indicates a problem,
  for example, "tech_alert_level": "warning"
- **Projects-New, Project-Update**
  Allows you to create a new project, give it a name, define the place of measurement, define setup of measuring devices etc.
- **Activate**
  Allows you to configure measuring devices for a given project.
- **Measurements-Start**
  Starts the measurement. The system starts recording data and sending it to Svannet.
- **Measurements-GetTimeHistData**
  Allows you to download a fragment of measurement data for a given time range.
- **Measurements-Stop**
  Stops the measurement.

A.7. DETAILED COMMAND DESCRIPTION

System-Login

Logging to the system, the system returns a "token" needed to use the other API methods.

Default settings of "username": "api", "password": "password"

Request:

```
{
  "Type" : "C1",
  "Request" : "System-Login",
  "Parameters" : {
    "username" : "api",
    "password" : "password"
  }
}
```

Response (if successful):

```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "Token": "73422296348059491020",
  "Data": [],
  "Errors": []
}
```

Response (if error):


```
{
  "Status": "ERROR",
  "StatusMsg": "authentication failed",
  "Token": "",
  "Data": [],
  "Errors": []
}
```


System-ChangePassword

Allows you to change the login password to the system

Request:


```
{
  "Type" : "C1",
  "Request" : "System-ChangePassword",
  "Parameters" : {
    "Token" : "613560937280929297",
    "newPassword" : "xyz"
  }
}
```


Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "Data": [],
  "Errors": []
}
```


System-GetStatus

Returns basic information about the system state. Requires no parameters.

Comment:

"Active_task" refers to a project that has been "activated". I.e. the system is configured to perform measurements according to the settings of the indicated project.

Request:


```
{
  "Type"       : "C1",
  "Request"    : "System-GetStatus",
  “Token”              : “8237825999749733321”,
  "Parameters" : {
  }
}
```


Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "MsgId": "OK",
  "MsgText": "OK",
  "MsgParameters": {},
  "Data": {
    "system_type": "SD310",
    "system_id": "SD 310 #3510",
    "core_version": "1.10.4 (2021.06.16 A4) MR",
    "time": "2021.06.17 08:00:23 Europe/London Greenwich Mean Time",
    "status": "measuring",
    "tech_alert_level": "ok",
    "mtime_elapsed_seconds": "7729",
    "mtime_elapsed": "2:08:49",
    "mtime_%": "",
    "active_task": {
      "project_id": 1071,
      "task_id": 1,
      "first_time_start": "2021.06.14 06:25:20"
    },
    "internetConnection": true,
    "gps": {
      "dateTime": "2021/06/17 07:00:23",
      "latitudeWGS84": "",
      "latitudeDir": "",
      "longitudeWGS84": "",
      "longitudeDir": "",
```

```
      "speed": "",
      "heading": "",
      "latitudeDec": 52.172936666666665,
      "longitudeDec": 21.163738333333338
    },
    "test_activeR": 2
  },
  "requestProcessingTime": "41",
  "Errors": []
}
```


Devices-GetList

Returns basic information about the state of the server and the state of connected devices. It does not require any parameters.

Request:


```
{
  "Type"       : "C1",
  "Request"    : "Devices-GetList",
  “Token”              : “8237825999749733321”,
  "Parameters" : {
  }
}
```


Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "Data": [
    {
      "id": "sd310-105205",
      "type": "SD 310",
      "serial_no": 105205,
      "firmware": "1.16",
      "core_version": "1.10.4 (2021.06.17 A1) MR",
      "status": "measuring",
      "disk_storage": [
        {
```

```
      "name": "SD card",
      "free_bytes": 19965165568,
      "total_bytes": 29528682496,
      "free_giga_bytes": 18.6,
      "total_giga_bytes": 27.5,
      "percentage_used": 33,
      "tech_alert_level": "ok"
    }
  ],
  "gps": {
    "dateTime": "2021/06/17 11:10:28",
    "latitudeWGS84": "5210.3835",
    "latitudeDir": "N",
    "longitudeWGS84": "02109.8334",
    "longitudeDir": "E",
    "speed": "0.00",
    "heading": "88.12",
    "latitudeDec": 52.17305833333333,
    "longitudeDec": 21.16388833333333
  },
  "tech_alert_level": "ok"
},
{
  "id": "extUnit-dust",
  "type": "dust meter",
  "label": "Dust M.",
  "status": "measuring",
  "status_msg": "measuring",
  "last_order": "start order completed",
  "serial_no": "...",
  "firmware": "?",
  "firmware_full": "?",
  "tech_alert_level": "ok",
  "current_results": [
    {
      "meas_type": "PM10",
      "value": 2,
      "unit": "µg/m³",
      "timestamp": "2021-06-17 12:10:29"
    }
```

```
    ],
    "messages": []
   }
 ],
 "Errors": []
}
```


Projects-GetList

Returns a list of projects in the form of a JSON array.

Request:


```
{
 "Type" : "C1",
 "Request" : "Projects-GetList",
 "Parameters" : {
 }
}
```


Response:


```
{
 "Status": "OK",
 "StatusMsg": "OK",
 "requestProcessingTime": "376",
 "Data": [
  {
   "id": 1097,
   "uid": "Test 101",
   "status": "",
   "time_created": "2021-06-18 09:15:37",
   "time_modified": "2021-06-18 09:16:50",
   "venue_gps_lat": "52.173108",
   "venue_gps_lon": "21.163992",
   "dev_settings_2": {
    "Dust": {
     "meas_type": "PM 10.0",
     "split_time": "3600",
     "logger_step": "30"
    },
    "Meteo": {},
    "Radar": {},
```

```
      "Sound": {},
      "Camera": {},
      "version": 2,
      "Schedule": {
        "units": null,
        "duration": null,
        "start_at": null
      }
    },
    "alarms": {
      "alarms": [
        {
          "type": "dust",
          "threshold": "2",
          "duration_secs": "2"
        }
      ]
    }
  },
  {
    "id": 1096,
    "uid": "tttest",
    "status": "cancelled",
    "time_created": "2021-06-09 13:10:27",
    "time_modified": "2021-06-09 13:14:04",
    "venue_gps_lat": "52.173080",
    "venue_gps_lon": "21.164073",
    ...
```


Projects-New

Request:


```
{
  "Type" : "C1",
  "Request" : "Projects-New",
  "Parameters" : {
    "uid" : "Project YYY",
    "descr" : "descr. of the Project ..."
  }
```

```
}
```


Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "requestProcessingTime": "615",
  "Data": [
    {
      "id": 1098,
      "uid": "Test 102",
      "status": "",
      "time_created": "2021-06-30 09:01:52",
      "time_modified": "2021-06-30 09:01:52",
      "venue_gps_lat": "52.173270",
      "venue_gps_lon": "21.163955",
      "dev_settings_2": {},
      "alarms": {}
    }
  ],
  "Errors": []
}
```


Projects-Update

Request:


```
{
  "Type" : "C1",
  "Request" : "Projects-Update",
  "Parameters" : {
    "id" : 1098,
    "descr" : "updated descr. of the project ...",
    "dev_settings_2": {
        "Dust": {
          "meas_type": "PM 1.0",
          "split_time": "600",
          "logger_step": "10"
        },
        "Schedule": {
          "units": null,
```

```
        "duration": null,
        "start_at": null
      },
      "version": 2
    }
  }
}
```


Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "requestProcessingTime": "102",
  "Data": [
    {
      "id": 1098,
      "uid": "Test 102",
      "status": "",
      "time_created": "2021-06-30 09:01:52",
      "time_modified": "2021-06-30 09:05:45",
      "venue_gps_lat": "52.173270",
      "venue_gps_lon": "21.163955",
      "dev_settings_2": {
        "Dust": {
          "meas_type": "PM 1.0",
          "split_time": "600",
          "logger_step": "10"
        },
        "version": 2,
        "Schedule": {
          "units": null,
          "duration": null,
          "start_at": null
        }
      },
      "alarms": {}
    }
  ],
  "Errors": []
}
```

Projects-Activate

After sending the Projects-Activate command the system status changes to "configuringDevices", System-Get Status returns "status": "configuringDevices", and when device configuration is finished System-Get Status will start returning "status": "idle" and you can activate the Measurements-Start command to start the measurement.

Request:


```
{
  "Type": "C1",
  "Request": "Projects-Activate",
  "Parameters": {
    "id": 1098
  }
}
```


Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "requestProcessingTime": "278",
  "Data": [],
  "Errors": []
}
```


Projects-Cancel

This command marks the project as "cancelled". In case of running out of disk space, the oldest project marked as “cancelled” will be deleted to make room for the current measurements.

Request:


```
{
  "Type" : "App-Request",
  "Request" : "Projects-Cancel",
  "Parameters" : {
    "id" : "1023"
  }
}
```


Response:


```
{
```

```
  "Status": "OK",
  "StatusMsg": "OK",
  "requestProcessingTime": "278",
  "Data": [],
  "Errors": []
}
```


Projects-Restore

This command removes the "cancelled" mark on the project.

Request:


```
{
  "Type" : "App-Request",
  "Request" : "Projects-Restore",
  "Parameters" : {
    "id" : "1023"
  }
}
```


Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "requestProcessingTime": "278",
  "Data": [],
  "Errors": []
}
```


Measurements-Start

Request:


```
{
  "Type": "C1",
  "Request": "Measurements-Start",
  "Parameters": {
  }
}
```

Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "requestProcessingTime": "278",
  "Data": [],
  "Errors": []
}
```


Measurements-GetTimeHistData

Request:


```
{
  "Type": "C1",
  "Request": "Measurements-GetTimeHistData",
  "Parameters": {
    "project_id": "1098",
    "time_from": "2021-05-06 07:41:00.0",
    "time_to": "2021-05-06 07:41:10.0"
  }
}
```


Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "project_id": "1098",
  "time_from": "2021-06-30 09:53:41",
  "time_to": "2021-06-30 09:53:51",
  "step": "1000",
  "Data": {
    "dust": [
      {
        "mpoint_id": "1",
        "step": 1000,
        "Data": [
          {
```

```
          "time": "2021-06-30 09:53:47",
          "value": "10"
        },
        {
          "time": "2021-06-30 09:53:48",
          "value": "9"
        },
        {
          "time": "2021-06-30 09:53:49",
          "value": "11"
        },
        {
          "time": "2021-06-30 09:53:50",
          "value": "11"
        },
        {
          "time": "2021-06-30 09:53:51",
          "value": "10"
        }
      ]
    }
  ],
  "meteo": []
 },
 "requestProcessingTime": "33",
 "Errors": []
}
```


Measurements-Stop

Request:


```
{
 "Type" : "C1",
 "Request" : "Measurements-Stop",
 "Parameters" : {

 }
}
```

Response:


```
{
  "Status": "OK",
  "StatusMsg": "OK",
  "requestProcessingTime": "63",
  "Data": [],
  "Errors": []
}
```


A.8. DESCRIPTION OF SOME OF THE SERVER'S RESPONSE FIELDS

active_task

tech_alert_level

A.9. EXAMPLE OF USE IN C#

```
String message = "the request body“; //{"Type":"C1", "Request":"Projects-New", …

String server = "127.0.0.1";
int port = 9876;

TcpClient client = new TcpClient();

if( client.ConnectAsync(server, port).Wait(2000) )
{
   Byte[] data_oryg = System.Text.Encoding.Default.GetBytes(message);
   Byte[] data_UTF8 = Encoding.Convert(Encoding.Default, Encoding.UTF8, data_oryg);

   // Get a client stream for reading and writing.
   // Stream stream = client.GetStream();

   NetworkStream stream = client.GetStream();
   stream.WriteTimeout = 1000;

   // Send the message to the connected TcpServer.
   stream.Write(data_UTF8, 0, data_UTF8.Length);
```

```
    Time_Start = DateTime.Now;

    //Receive the TcpServer.response.
    byte[] myReadBuffer = new byte[1024];
    StringBuilder myCompleteMessage = new StringBuilder();
    int numberOfBytesRead = 0;

    do
    {
      numberOfBytesRead = stream.Read(myReadBuffer, 0, myReadBuffer.Length);

      myCompleteMessage.AppendFormat("{0}",
                                     Encoding.UTF8.GetString(myReadBuffer, 0, numberOfBytesRead));

      Thread.Sleep(100);

    }
    while (stream.DataAvailable);

    String responseData = String.Empty;
    responseData = myCompleteMessage.ToString();

    stream.Close();
    client.Close();
}
```

Appendix B. DATA FILE STRUCTURES

B.1. GENERAL STRUCTURE OF THE SD 310 FILE

Each file containing data from the **SD 310** instrument consists of the following elements:

- the SvanPC file header (see Table B.1)
- JSON header length (see Table B.2)
- Unit & measurement parameters in JSON format (see Table B.3)
- contents of the file from the logger (see Table B.16)

Table B.1. SvanPC file header

Word number	Name	Comment
0..2	“SvanPC”	text (id)
3	0x001A	id
4	0x0020	header length (32)
5..6	reserved	
7..8	reserved	
9	0x0002	flag indicating json format of headers
10..15	reserved	

Table B.2. JSON header length

Word number	Name	Comment
0..1	JSONLength	json header length in bytes (32-bit)

Table B.3. Unit & measurement parameters (JSON format)

Name	Type	Comment
Filename	string	original filename
FileDateTime	string	file creation date & time (ISO format YYYY-MM-DDTHH:MM:SS)
UnitType	number	type of the unit: 310
UnitSubtype	number	subtype of the unit: 1 – SD310
UnitNumber	number	unit number
SoftwareVersion	number	software version (value*100)
FilesystemVersion	number	file system version (value*100)
MeasureStart	string	date & time of measurement start (ISO format YYYY-MM-DDTHH:MM:SS)
MeasureStartMS	number	milliseconds of measurement start
LoggerStep	number	logger step in milliseconds

LoggerRecord	object	logger record parameters (c.f. Table B.4)
Meteo	object	meteo parameters (c.f. Table B.2)
Dust	object	dust parameters (c.f. 0**Error! Reference source not found.**)
Radar	object	radar parameters (c.f. Table B.6)

Table B.1. LoggerRecord

Name	Type	Comment
RecSize	number	logger record size in bytes (whole record including Meteo, Dust, …)
TimestampSec	bool	record contains timestamp in seconds
TimestampMS	bool	record contains timestamp in milliseconds
StepSec	bool	record contains step in seconds
StepMS	bool	record contains step in milliseconds

Table B.2. Meteo parameters

Name	Type	Comment
UnitType	number	type of the meteo unit
UnitNumber	number	unit number
SoftwareVersion	string	software version
IntPeriod	number	integration period in milliseconds
DistributionN	number	length of distribution vectors
MeteoRecord	object	results saved in record (c.f. Table B.3)

Table B.1. Meteo record

Name	Type	Comment
RecSize	number	meteo record size in bytes
Temperature	bool	temperature
Pressure	bool	pressure
Humidity	bool	humidity
AvgWindSpeed	bool	average wind speed
WindDirection	bool	wind direction
MaxWindSpeed	bool	maximum wind speed
WindDirTotalPuffs	bool	wind direction total samples
RainFlag	bool	rain flag
RainIntensity	bool	rain intensity
RainAccumulation	bool	rain accumulation
RainDuration	bool	rain duration
Dewpoint	bool	dew point
WindDirDistribution	bool	wind direction distribution vector
MaxWindDistribution	bool	maximum wind speed distribution vector
AvgWindDistribution	bool	average wind speed distribution vector

Table B.2. Dust parameters

Name	Type	Comment
UnitType	number or string	type of the dust unit
UnitNumber	number	unit number
SoftwareVersion	string	software version
IntPeriod	number	integration period in milliseconds
DustRecord	object	results saved in record (c.f. Table B.3)

Table B.1. Dust record

Name	Type	Comment
RecSize	number	dust record size in bytes
Status	bool	status
PM1	bool	PM1
PM2.5	bool	PM2.5
PM10	bool	PM10
TSP	bool	TSP
Temperature	bool	temperature
Pressure	bool	pressure
Humidity	bool	humidity
AirFlow	bool	air flow

Table B.2. Radar parameters

Name	Type	Comment
UnitType	string	type of the radar unit (e.g. Wavetronix XXX)
DateTime	string	radar date & time (ISO format YYYY-MM-DDTHH:MM:SS)
Lines	array of objects	array of line definition (c.f. Table B.3)
Approaches	array of objects	array approach definition (c.f. 0)
Lengths	array of objects	array of length definition (c.f. Table B.3)
Speeds	array of objects	array of speed definition (c.f. Table B.4)
Directions	array of objects	array of direction definition (c.f. Table B.5)
RadarRecord	object	results saved in record for each line (c.f. Table B.6Table B.1)

Table B.1. Radar line definition

Name	Type	Comment
Description	string	line description
Direction	number	line direction (0 – right, 1 – left)

Table B.2. Radar approach definition

Name	Type	Comment
Description	string	approach description
Directions	array of numbers	approach directions (0 – right, 1 – left)
LanesPerApproach	array of numbers	lanes per approach
LaneAssignments	array of numbers	approach lane assignments

Table B.3. Radar length definition

Name	Type	Comment
Boundary	number	length boundary [m]

Table B.4. Radar speed definition

Name	Type	Comment
Boundary	number	speed boundary [km/h]

Table B.5. Radar direction definition

Name	Type	Comment
BinByDirection	number	0 – disabled, 1 – enabled

Table B.6. Radar record

Name	Type	Comment
RecSize	number	radar record size in bytes
AvgSpeed	bool	average speed
Volume	bool	volume
AvgOccupancy	bool	average occupancy
Speed85	bool	average speed for 85% of vehicles
Headway	bool	headway
Gap	bool	gap
LengthBins	bool	length bins
SpeedBins	bool	speed bins
DirectionBins	bool	direction bons

Table B.7. Contents of the logger file

Word number	Name	Comment
...	Record0	Table B.8
…	Record1	Table B.1
…	…	…

Table B.1. Record format

Word number	Name	Comment
+2	TimestampSec	time of record in seconds (presence depends on LoggerRecord flags)
+4	TimestampMS	time of record in milliseconds (presence depends on LoggerRecord flags)
+2	StepSec	step of record in seconds (presence depends on LoggerRecord flags)
+4	StepMS	step of record in milliseconds (presence depends on LoggerRecord flags)
…	Meteo record	Table B.2
…	Dust record	Table B.2
…	Radar record	Table B.3

Table B.1. Meteo record

Word number	Name	Comment
+1	MeteoTemperature	temperature [0.1°C] (presence depends on MeteoRecord flags)
+1	MeteoPressure	pressure [hPa] (presence depends on MeteoRecord flags)
+1	MeteoHumidity	humidity [0.1%] (presence depends on MeteoRecord flags)
+1	MeteoAvgWindSpeed	average wind speed [0.1m/s] (presence depends on MeteoRecord flags)
+1	MeteoWindDirection	wind direction (0xFFFF => undefined, presence depends on MeteoRecord flags)
+1	MeteoMaxWindSpeed	max wind speed [0.1m/s] (presence depends on MeteoRecord flags)
+2	WindDirTotalPuffs	wind direction samples (presence depends on MeteoRecord flags)
+1	Rain	rain flag (presence depends on MeteoRecord flags)
+1	RainIntensity	rainfall intensity [0.1mm/h] (presence depends on MeteoRecord flags)
+2	RainAccumulation	sum of rainfall [0.01mm] (presence depends on MeteoRecord flags)
+2	RainDuration	duration of precipitation in seconds (presence depends on MeteoRecord flags)
+1	Dewpoint	dewpoint [0.1°C] (presence depends on MeteoRecord flags)
+DistributionN	WindDirDistribution	wind direction distribution vector [0.1%] (presence depends on MeteoRecord flags)
+DistributionN	MaxWindDistribution	max wind speed distribution vector [0.1m/s] (presence depends on MeteoRecord flags)
+DistributionN	AvgWindDistribution	average wind speed distribution vector [0.1m/s] (presence depends on MeteoRecord flags)

Table B.2. Dust record

Word number	Name	Comment
+1	DustStatus	sum of the following flags: b0 – Sensor Calibration Error: Zero reading to low b1 – Sensor Calibration Error: Zero reading to high b4 – IOP Error (Laser) b5 – Counter Error (Sensor) b6 – Flow Regulation Error b12 – PM1 result calculated. b13 – PM2.5 result calculated. b14 – PM10 result calculated. b15 – TSP result calculated. If bits 0 and 1 are set simultaneously, this means Sensor Calibration Error: Stability error, too many retries (presence depends on DustRecord flags)
+2	DustPM1	PM1 in µg/m^3 (presence depends on DustRecord flags & valid only if bit 12 in DustStatus is set)
+2	DustPM2.5	PM2.5 in µg/m^3 (presence depends on DustRecord flags & valid only if bit 13 in DustStatus is set)
+2	DustPM10	PM10 in µg/m^3 (presence depends on DustRecord flags & valid only if bit 14 in DustStatus is set)
+2	DustTSP	TSP in µg/m^3 (presence depends on DustRecord flags & valid only if bit 15 in DustStatus is set)
+1	DustTemperature	temperature [0,1°C] (presence depends on DustRecord flags)
+1	DustPressure	pressure [hPa] (presence depends on DustRecord flags)
+1	DustHumidity	humidity [0.1%] (presence depends on DustRecord flags)
+1	DustAirFlow	air flow [0.1lpm] (presence depends on DustRecord flags)

Table B.3. Radar record

Word number	Name	Comment
…	Line1	line 1 results (c.f. 0)
…	Line2	line 2 results (c.f. 0)
…	…	…
…	LineN	line N results (c.f. 0)

Table B.4. Radar line record

Word number	Name	Comment
+1	AvgSpeed	average speed [0.1 km/h] (presence depends on RadarRecord flags)
+2	Volume	number of vehicles
+1	AvgOccupancy	average occupancy [0.01%] (presence depends on RadarRecord flags)
+1	Speed85	average speed for 85% of vehicles [0.01%] (presence depends on RadarRecord flags)
+2	Headway	headway [ms] (presence depends on RadarRecord flags)
+2	Gap	gap [ms] (presence depends on RadarRecord flags)
+2*NofLengths	LengthBins	length bins (presence depends on RadarRecord flags)
+2*NofSpeeds	SpeedBins	speed bins (presence depends on RadarRecord flags)
+2*NofDirections	DirectionBins	direction bins (presence depends on RadarRecord flags)

Appendix C. SD 310 TECHNICAL DATA[2]

No.	*Parameter*	*Value/ Description*
1. Physical data		
1	Dimensions	360 x 200 x 90 mm
2	Weight	3 kg
3	Ingress Protection Rating	IP 65
4	Working temperature range	-20°C do +60°C (Ambient air temperature, without direct sunlight). *Note: Outside this range the device will automatically switch itself off.*
5	Storage temperature range	-40°C to +70°C
6	Working relative humidity range	up to 99 %RH
2. Power Supply		
1	Built-in AC power supply	input 90 ÷ 305 VAC output 40W, +15 VDC, 2.67 A
2	External DC source (not included)	voltage range 10.5 ÷ 17 V DC, e.g. 12 V accumulator
3. Communication Interfaces		
1	Cellular modem	4G LTE Huawei E3372h-153
2	WLAN / Bluetooth®	The SD 310 monitoring systems controller includes Raspberry Pi 3 Model B+ single board computer with built-in CYW43455 radio chip and microstrip PCB antenna. The CYW43455 is a single chip integrating 802.11 WLAN and Bluetooth® which adds wireless connectivity to the SD 310 device. Implemented by SD 310 IEEE 802.11x features of the chip are: • Standards supported: IEEE 802.11b/g/n • Frequency range: 802.11b/g/n – 2.400-2.500 GHz

[2] Our Company's policy is based upon continuous product development and innovation. Therefore, we reserve the right to change the specifications without any prior notice whatsoever

No.	*Parameter*	*Value/ Description*
		• Channels: 802.11b/g/n – 1-13 (Europe), 1-14 (USA) • Supported data rates: 802.11b – 1, 2, 5.5, 11 Mbps 802.11g – 6, 9, 12, 18, 24, 36, 48, 54 Mbps 802.11n – MCS0-MCS7, HT20 (6.5-65 Mbit/s) • Supported bandwidth: 802.11n – 20/40 MHz • Receiver Minimum Input Level Sensitivity: 802.11b (1 Mbps DSSS, 8% PER for 1024 octet PSDU) -98.7 dBm 802.11b (11 Mbps DSSS, 8% PER for 1024 octet PSDU) -90.7 dBm 802.11g (6 Mbps OFDM, 10% PER for 1024 octet PSDU) -95.3 dBm 802.11g (54 Mbps OFDM, 10% PER for 1024 octet PSDU) -78.2 dBm 802.11n (MCS0, 10% PER for 4096 octet PSDU) -94.8 dBm 802.11n (MCS7, 10% PER for 4096 octet PSDU) -74.7 dBm • Transmitter output power (max): (DSSS/CCK) 20.5 dBm (OFDM, BPSK) 20.0 dBm (OFDM, 64QAM) 19.0 dBm • Security protocols: AES and TKIP, CCX (v2.0, v3.0 and v4.0), WPS (WLAN Protected Setup), WPA/WPA2 (PSK) encryption Implemented by SD 310 Bluetooth® features of the chip are: • Bluetooth® Low Energy v4.2 • Operating frequency range: 2.402 – 2.480 GHz • Transmit power (max): 12 dBm (GFSK, TX power at Bluetooth) • Sensitivity: -93.5 dBm (GFSK, 0.1% BER, 1 Mbps) • Channels: 40 • Modulation: GFSK Notes: 5GHz bandwidth operation (802.11a/n) of the CYW43455 chip is intentionally switched off by the SD 310 software. Raspberry Pi 3 Model B+ certifications • Radio & EMC: - EU: 2014/53/UE RED (Radio Equipment Devices) Directive - USA: FCC Approval; (FCC ID: 2ABCB-RPI3BP) - Canada: IC Approval; (IC: 20953-RPI3P) • Safety: - 2014/53/UE RED (Radio Equipment Devices) Directive

No.	*Parameter*	*Value/ Description*
		THIS DEVICE COMPLIES WITH PART 15 OF THE FCC RULES. OPERATION IS SUBJECT TO THE FOLLOWING TWO CONDITIONS: (1) THIS DEVICE MAY NOT CAUSE HARMFUL INTERFERENCE, AND (2) THIS DEVICE MUST ACCEPT ANY INTERFERENCE RECEIVED, INCLUDING INTERFERENCE THAT MAY CAUSE UNDESIRED OPERATION
3	LAN network	Gigabit Ethernet
4	Ports / interfaces	up to 4 USB Host ports up to 3 RS 232 interfaces
4. Other		
1	Memory	32 GB (microSD card)
2	GPS	The instrument has a built-in A2235-H GPS module of Maestro Wireless Solutions (HK) Ltd. intended for logging position and time definition. GPS is an antenna module with SiRF Star IV ROM based chip and an on-board integrated antenna. • Frequency: L1 (= 1,575 MHz) • Position Accuracy (horizontal): < 2.5 m CEP (autonomous), • Tracking Sensitivity: -163dBm • Time accuracy: <1µs (directly depends on position deviation)
3	External Connectors	3-pin AC mains connector 8-pin External Interface port, incl. RS232 interface and power output 15 V, 1 A

Appendix D. SUPPORTED DEVICES

D.1. ES-642 – REMOTE DUST MONITOR

ES-642 Remote Dust Monitor is an industrial air-quality sensor designed to provide real time particle concentration measurements in both indoor and outdoor environments. It is connected to SD 310 via serial RS 232C interface.

ES-642 measures particulate concentration using a highly sensitive forward scatter laser nephelometer, which has a measurement range of 0 to 100 mg/cubic meter (0 to 100,000 µg/cubic meter). As supplied, the ES-642 provides TSP particulate monitoring. Optional sharp-cut cyclones are available for PM1, PM2.5 or PM10.

ES-642 transfers measured data to SD 310 every second.

Note: *For installation, setup, and field calibrations please refer the ES-642 manual (https://metone.com/products/es-642/).*

Note: *SD 310 saves results from weather or dust monitor stations in the logger file with the extension SVL.*

D.2. SP 276 – WEATHER STATION

SP 276 is a GILL GMX600 type weather station used with SD 310. It is connected to SD 310 via serial RS 232C interface.

SP 276 measures 6 most essential weather parameters (barometric pressure, humidity, precipitation, temperature, wind speed and direction). It is compact and light-weight, has no moving parts and can be easily installed with a one-bolt mounting method.

All measurement weather parameters are transferred from SP 276 to SD 310 every second.

Note: *If your GILL weather station is equipped with the wind sensor, then it is critical to set the correct sensor orientation. The North direction is marked at the bottom of the weather station. Use real-life compass or mobile app to determine North direction.*

Note: *See also GILL GMX600 User Guide.*

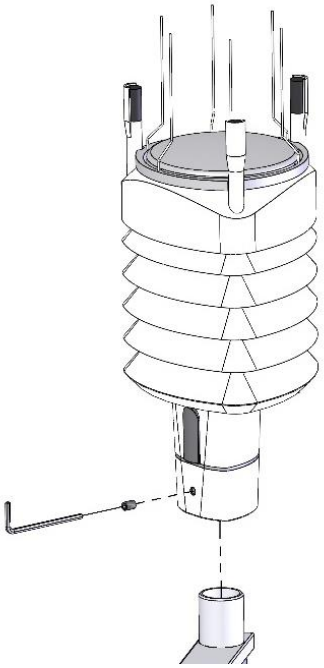
D.3. SP 275 – WEATHER STATION

SP 275 is a Vaisala Weather Transmitter WXT5xx type meteorological station used with SD 310. It is connected to SD 310 via serial RS 232C interface.

SP 275 measures 6 most essential weather parameters (barometric pressure, humidity, precipitation, temperature, wind speed and direction) and hail intensity. It is compact and light-weight, has no moving parts, has internal heating and can be easily installed with a one-bolt mounting method.

SP 275 has an automatic control circuit that switches the heating on at low temperatures.

Weather measurements results are transferred from SP 275 to SD 310 every second, but precipitation is sent every 10 seconds intervals.



Note: *If your weather station is equipped with the wind sensor, then it is critical to set the correct sensor orientation. The North direction is marked at the bottom of the weather station. Use real-life compass or mobile app to determine North direction.*

Note: *See also Vaisala WXT5xx User Guide.*