Watcher Manual

Retroview

Flussonic

Table of contents

1. Products 3

1.1 Flussonic Watcher 3

2. Manuals & Guides 7

2.1 Watcher Quick Start Guide 7

2.2 Developers 12

3. Solutions 14

3.1 Advertising on Cameras 14

3.2 Embedding cameras on your website 16

3.3 Search the archive for visitors 19

3.4 How to Provide RTSP Access for an External System 21

3.5 Viewing Watcher cameras on SMART TV, STB and IPTVPORTAL applications 22

3.6 Using Zones to Distribute Cameras 23

4. Administration Guides 24

4.1 Selecting hardware 24

4.2 Installation 31

4.3 Managing cameras and permissions 40

4.4 Video analytics 107

4.5 Watcher Settings 127

4.6 Using Flussonic Agent 142

4.7 Mobile Apps 145

4.8 VSAAS.IO video surveillance service 148

5. User Guides 164

5.1 Watcher user interface 164

5.2 My Cameras 166

5.3 Adding a camera to Favorites 174

5.4 Viewing mosaics 175

5.5 Map 177

5.6 Events 178

5.7 Profile 179

6. Developers Guides 180

6.1 API 180

1. Products

1.1 Flussonic Watcher

Flussonic Watcher is a complete video surveillance software system that works with a distributed IP camera network. This system can be used for video streaming, recording and managing video archives.

Flussonic Watcher is a scalable and ready-to-use system with flexible integration options. This web and mobile-oriented solution can solve various business tasks: from launching a corporate cloud video monitoring to a municipal video surveillance system that covers the entire city or state.

Watcher supports small to medium to large projects with unlimited number of cameras. The number of cameras is limited only by hardware.

When working in a cluster of servers, Watcher ensures the fault-tolerance of stream ingest (failover).

1.1.1 Areas of appliance

- Internet providers—to launch a dedicated cloud-based video surveillance service and offer it to their customers for an additional fee. Watcher can be easily integrated with any provider and its existing billing system.
- Management companies—for live video broadcasts from socially significant facilities and construction monitoring.
- Production—to perform audio-video monitoring on factories, courts, polling stations, etc.
- Municipal and federal projects—to provide free access to public cameras, as well as limited access to security organizations.

1.1.2 Flussonic Watcher features

- **Certified in-house cameras**. Wide model range of cameras with high quality of assembly and tactical-technical characteristics (TTC). Test cameras before buying. Audio listening. Access to shared cameras. Public access to video cameras.
- **Agent**
- **Single provider of video cameras and software**. Guaranteed hardware and software compatibility.
- **Connect any video camera via RTSP**.
- **Bilateral audio communication**
- **Mainstream and sub-stream recording**.
- **USB license**. Software continues to work when disconnected from the Internet.
- **NVR**
- **Video analytics**. Face recognition, license plate recognition, detection of special vehicles and vehicles without license plates. Video analytics on NVR. Video analytics notifications. Video analytics reports. Communication of video analytics with access control system.
- **Event notifications: line crossing, movement in the zone, masking**. Subscriber event notifications.
- **API for integration with billing**.
- **SDK for Android and iOS**. Video embedding in your mobile app.
- **STARVIS**. Fixing object color at night.
- **Web interface**.
- **Android and iOS mobile app**.
- **Branding**. Branding of web interface and mobile application on Android and iOS.
- **Logging of user actions**
- **Public documentation**
- **Compatibility with popular protocols and codecs**
- **Proprietary authorization system**
- **DVR RAID**. Archive replication. Archive export.

- **DVR web player**
- **Scaling**
- **Project support**

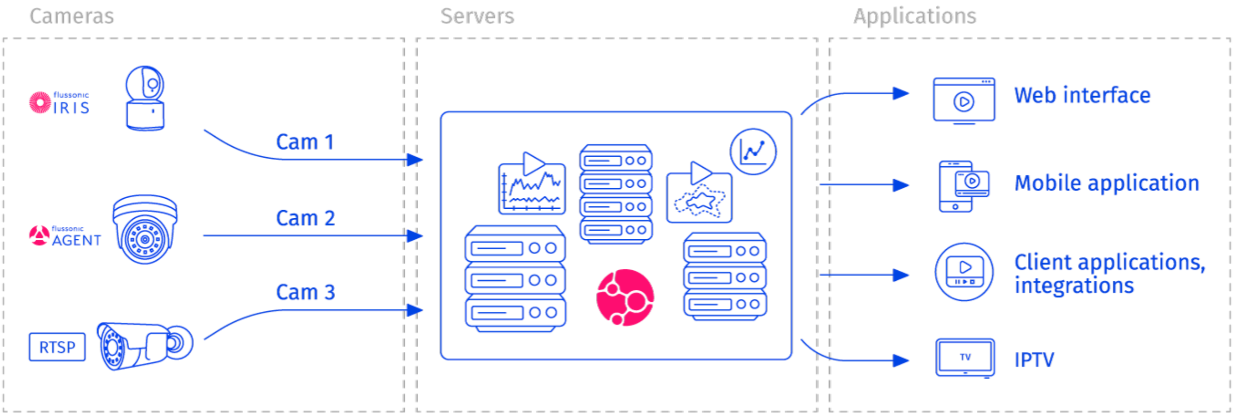
1.1.3 Start working with Flussonic Watcher

To start working with Flussonic Watcher, do the following:

1. Request a trial by filling in the trial form on our website
2. Follow the steps in the Watcher quick start guide.

1.1.4 Parts of Flussonic Watcher

The general Flussonic Watcher architecture is shown below:



Details on the system parts:

Cameras

- Any camera with **RTSP** support can be added to Watcher. However, connection without special tools (described below) may lead to some limitations.
- Cameras with **Agent** are a plug-and-play solution designed to make the connection process simple and secure by creating an encrypted tunnel between the server and the camera. Please refer to this page for details.
- Cameras with our in-house developed **Iris** firmware bring all the advantages of the Agent while providing additional features to connect a Wi-Fi camera via a QR code in just a few simple steps. Please refer to this page for details. You can order the cameras with Iris on our website.

Servers

Watcher includes the following types of servers:

- **Streamers** with robust and reliable FlussoniC Media Server providing:
  - Capturing video from cameras.
  - Audio transcoding.
  - Recording the archive.
  - Multi-protocol video delivery to various devices.

> **Note**
>
> When Media Server runs as part of Watcher, its configuration is controlled by Flussonic Central via config_external. Changing the stream configuration directly in the Media Server web interface or configuration file is not supported.

- **Flussonic Central** provides:
  - Camera and Agent provisioning and activation, distribution between streamers.
  - Load balancing, failover and scalability.
  - Remote camera setup and maintenance.
- **Flussonic Watcher VMS** is a managing server providing:
  - User access management.
  - Archive quotas.
  - Camera hierarchy managing.

  Watcher VMS can be installed separately from the rest of the system components on one or more separate servers to distribute the load and improve reliability.
- **Flussonic Vision** is a server with video analytics modules installed providing:
  - Video analytics for online face recognition and LPR.

FLUSSONIC WATCHER EDITIONS

Flussonic Watcher comes in two editions that define servers configuration:

1. **Single** is a basic version limited to one server for all the above functions. It is good for smaller projects not requiring failover or branding, for example for video surveillance on one site.
2. **Cluster** is a scalable edition with failover capabilities and out-of-the-box branding tools with different functions distributed between several servers. It is good for VSaaS providers and large-scale video surveillance systems.

Hardware requirements to servers depend on the edition you select.

You may also find general info and instructions on server calculations in Selecting hardware for video surveillance.

Applications

- Both users and administrators will appreciate the comprehensive ready-to-use web interface available in any modern mobile or desktop browser. The web interface includes a dashboard with cameras and archive, favorites, map, user and camera management tools, system settings, interface branding tools and much more.
- Let your users access their cameras and archive from anywhere anytime with Watcher iOS and Android mobile applications: all they need is a mobile device. In the basic version, users need to know a few numbers (Operator ID) to log into the app and view the cameras. We also offer a service for mobile applications branding: we will design our application in your company style (with your logo and color scheme) and hide the ID. Please contact our support team for details.

- Use Watcher API for integration with complex solutions requiring video surveillance or create your own applications. You may also need the API to fine-tune the billing preferences in your VSaaS.
- User experience in Watcher is not limited with just web UI and mobile phones: you can watch video from cameras on a TV, as well as embed video on your website or share it with someone else using a direct link.

2. Manuals & Guides

2.1 Watcher Quick Start Guide

This quick start guide will teach you how to do the most common tasks in Flussonic Watcher. After looking through this guide you will be able to:

- Install Flussonic Watcher on one server (in Single mode).
- Add a virtual camera.
- View the archive records.
- Connect a mobile app.

You will need the following assets to follow the instructions below:

- A computer with Linux operating system and access to the Internet. See here for detailed system requirements.
- A license key. Contact our manager to purchase a license.

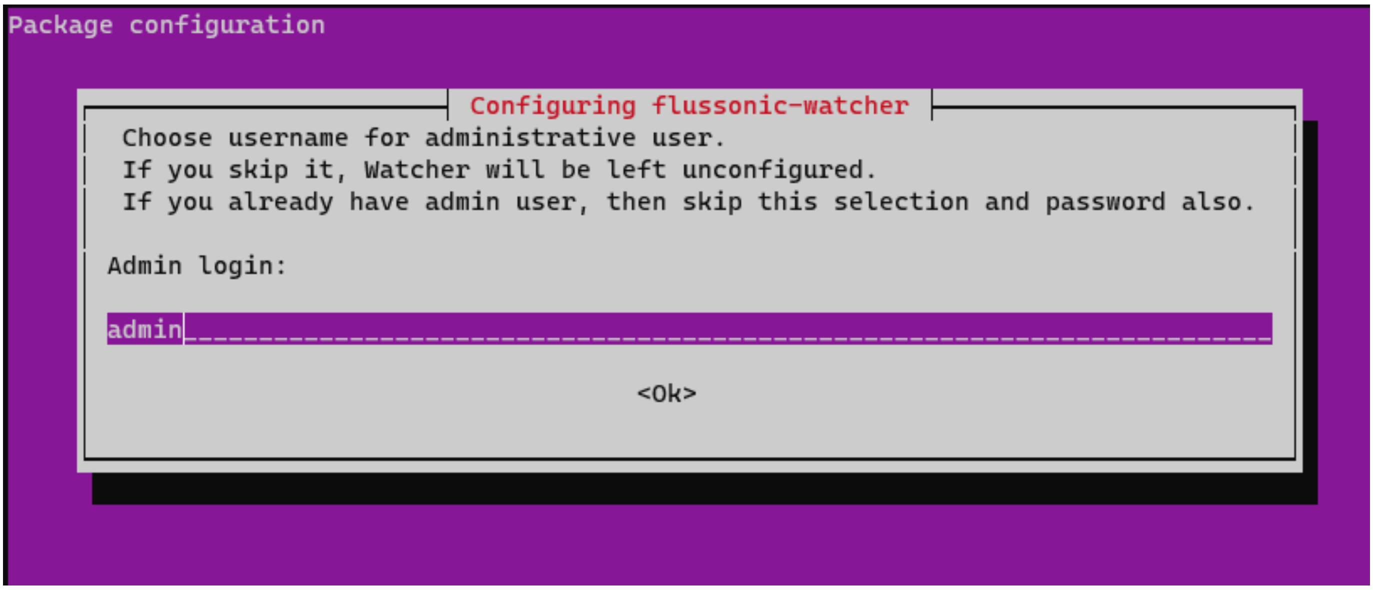
2.1.1 Installing Flussonic Watcher

Install Flussonic Watcher as follows:

1) On the server where you plan to run Watcher execute the commands:

wget -q -O /etc/apt/trusted.gpg.d/flussonic.gpg http://apt.flussonic.com/binary/gpg.key; echo "deb http://apt.flussonic.com binary/" > /etc/apt/sources.list.d/flussonic.list; apt update; apt install flussonic-watcher;

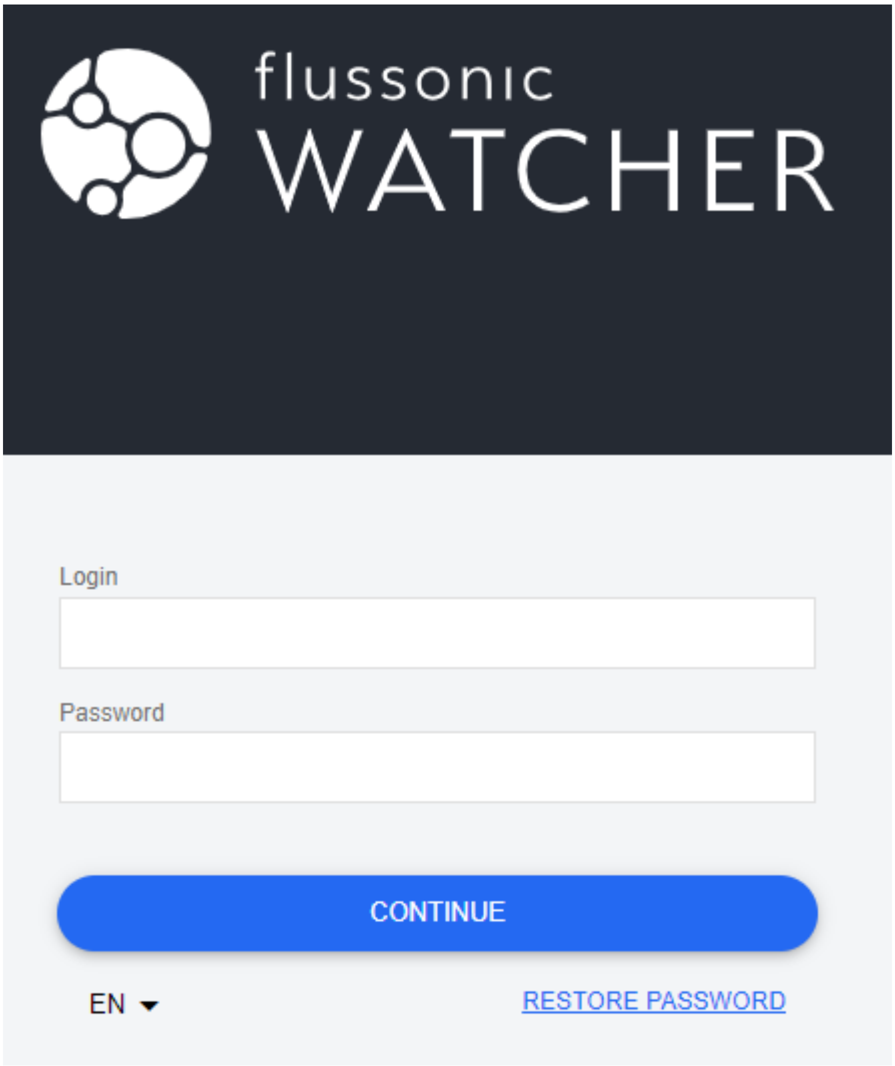
2) The installation wizard will prompt you to enter the necessary data: administrator's login and password, license key, and the path to the DVR where the archive will be stored.



```
Package configuration

                    Configuring flussonic-watcher
  Choose username for administrative user.
  If you skip it, Watcher will be left unconfigured.
  If you already have admin user, then skip this selection and password also.

 Admin login:

 admin_______________________________________________________________

                                <Ok>
```

Enter the requested information and wait for the installation to complete.

3) Open Watcher web UI at `http://(Watcher server address)`. Use login and password you set at the installation.



The installation for single-server mode is now complete. Please refer to Installing Watcher Cluster or Single for full installation procedure.

2.1.2 Adding a virtual camera

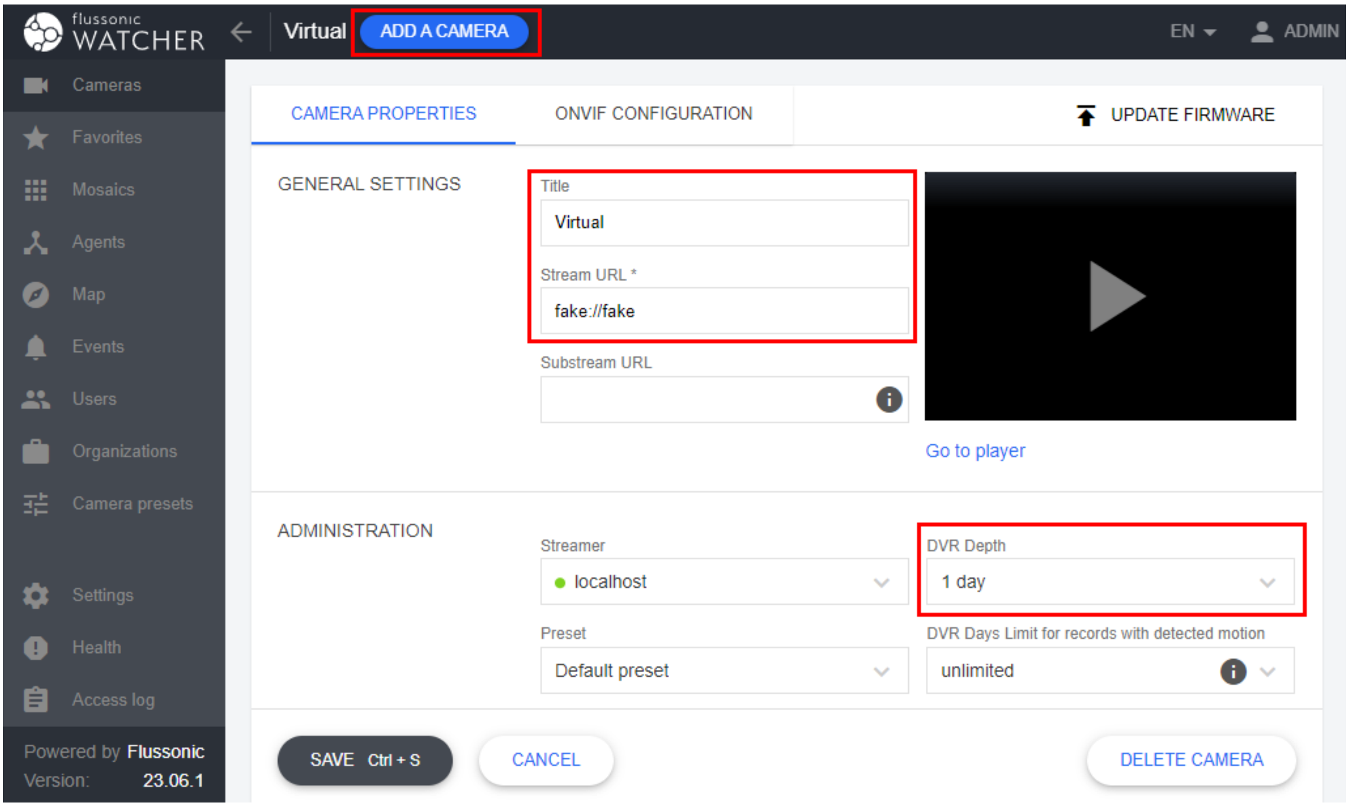
A virtual camera is a software stream that you can add as a camera to Watcher using `fake://fake` URL. Such a camera can be used for test purposes to check the settings, video and sound without a real camera.

To add a virtual camera:

1. Go to **Cameras** and click **Add camera**.
2. In the form that opens, enter the camera name and URL `fake://fake`.
3. Select some value in the **DVR Depth** drop-down list to enable the archive recording.
4. Click **Save**.

   After saving the settings, a screenshot of the camera will be displayed on the right. If the screenshot does not appear, refresh the page.

   Here you can also view live video or open it in full page using the **Go to player** button.



Title: Virtual

Stream URL *: fake://fake

Substream URL:

Streamer: localhost

DVR Depth: 1 day

Preset: Default preset

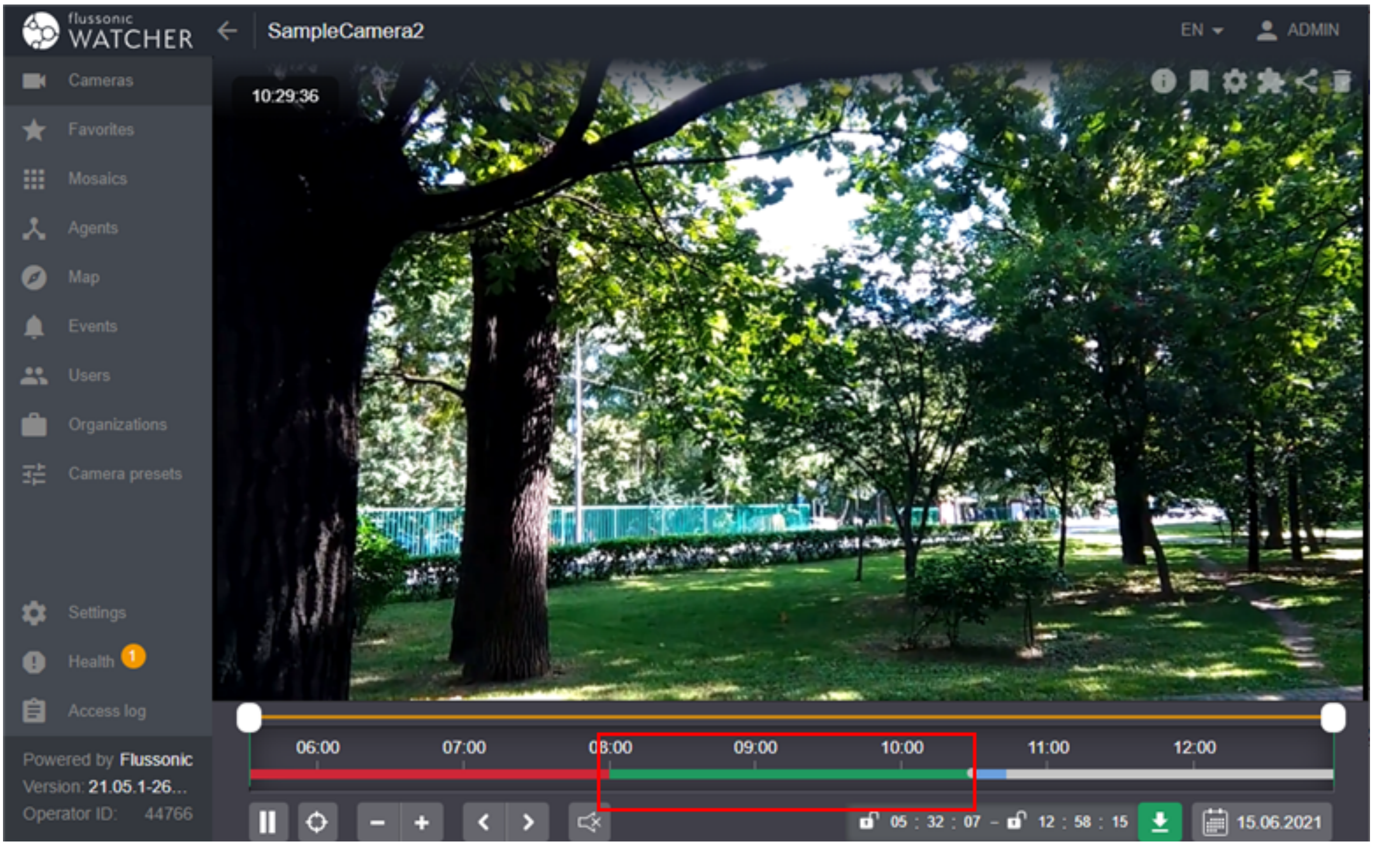
DVR Days Limit for records with detected motion: unlimited

Adding a virtual camera is completed. Please refer to Adding cameras for other ways of adding cameras.

2.1.3 Viewing the archive

To view the archive, go to the **Cameras** page and click on the tile of the desired camera. The player will open in live view mode.

You will see green areas on the timeline if the camera has an archive. Click the desired point in the green area to start archive playback from the selected moment.



Learn more about the player and playback controls here.

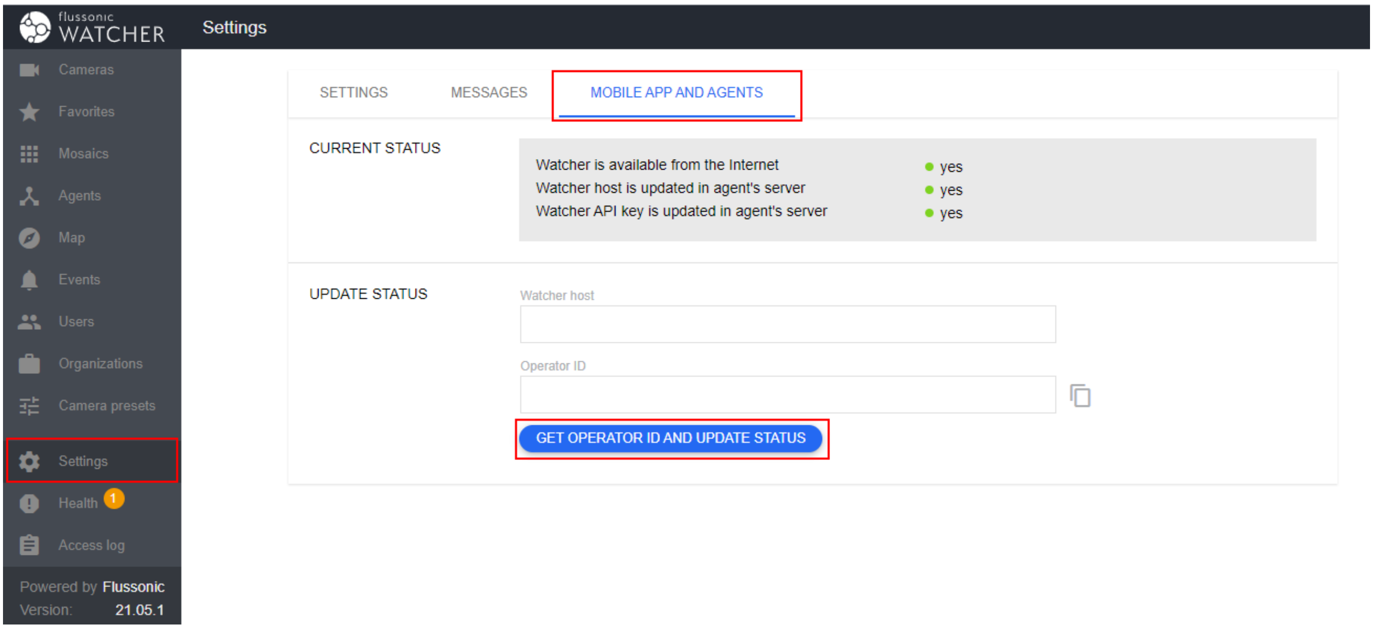
2.1.4 Connecting a mobile app

There are two types of mobile applications for Watcher:

- Traditional mobile applications that are installed from **Apple Store** or **Google Play**. These mobile apps are useful for adding cameras.
- Progressive Web Application (PWA) that is installed by **Add to Home Screen** option in the browser menu. This app provides all the functions that are available in the web UI.

You will need the Operator ID to log into the traditional app. Operator ID is a unique identifier of your Watcher. To get it:

1. In the web browser, go to your Watcher's public (external) URL or IP address accessible from the Internet. Do not use the server's local IP address because the Operator ID will not be issued and the Watcher mobile app will not connect in such case.
2. Log in to Watcher as an administrator
3. Go to **Settings** - **MOBILE APP AND AGENTS** and click **GET OPERATOR ID AND UPDATE STATUS**.
4. As a result, your `Operator ID` will appear in the **Operator ID** field.



CURRENT STATUS

Watcher is available from the Internet ● yes
Watcher host is updated in agent's server ● yes
Watcher API key is updated in agent's server ● yes

UPDATE STATUS

Watcher host

Operator ID

With that Operator ID, you can log into the mobile application and see the virtual camera you added earlier.

2.2 Developers

2.2.1 External video analytics integration

Flussonic Watcher allows for the integration of third-party video analytics. This enables:

- Storage of video episodes with desired events using the standard DVR mechanism.
- Search, view, and export external events alongside standard Watcher analytics events.
- Transmission of external events to other systems on par with other data.

Overall, events from third-party analytics are integrated on an equal footing with internal events.

To achieve this, it is necessary to:

- Retrieve the video stream via RTSP (if the analytics are indeed based on video and not the operation of another sensor).
- Identify the required events within the video.
- Based on the timestamps in RTSP, send **episodes** containing these events to Watcher.

RTSP for video analytics

Video can be retrieved using various protocols, but the RTSP protocol supports absolute time addressing, suitable for frame identification, and is the standard for IP video surveillance; hence, we recommend its use.

Our experience with RTSP from surveillance cameras indicates that the accuracy of this protocol suffices for frame-by-frame synchronization of different cameras.

To allow the analytics system to retrieve video, an API token must be obtained for Watcher.

Episodes

Watcher utilizes the data model **episode** instead of **event** for analytics events, and this distinction is crucial.

An episode has the following important characteristics:

- The identifier is generated by the source. You need to generate an ID yourself and use it, which is no more complicated than a uuid.
- An episode has a flexible start and end. This approach is more natural for video than a discrete event.
- You can adjust the boundaries of an episode if you manage to refine it further in the process.

The lifecycle of an episode is as follows:

- When analytics first detects something useful in the image at moment T1, an episode is created with a generated `episode_id`, `media` equal to the stream name, `opened_at=T1`, and `updated_at=T1`. The `opened_at` field is set once and then **never changes**.
- The created episode is immediately sent to Watcher via the `episode_save` call. Since the ID is generated by the source, there is no separate `episode_create` method.
- If the object detection extends over time, then `updated_at` in the episode is updated, which must always increase with any change in the episode. Otherwise, Watcher will ignore the changes.
- If it is conclusively known that the event registration in the frame has ended (for example, a car has left the scene), then the `closed_at` field can be set, and it **cannot be changed** afterwards.
- The `preview` field of an episode is for providing an exact screenshot of the detection, possibly cropped to the required dimensions. Feel free to update it if you manage to get a better picture. This field will be displayed in Watcher in the list of detections.
- The entire episode **will be overwritten each time**, so send all fields during updates. This isn't a partial update but a complete rewrite of the record.

Snowflake ID

It appears you might have intended to describe the structure of the 64-bit identifier you use instead of a UUID but didn't provide the details of its structure. Could you please provide the specifics of how this 64-bit identifier is constructed or any other details you need assistance with?

42 bits	10 bits	12 bits
millisecond timestamp	machine-id	sequence

The significance of using a 64-bit identifier that exceeds JavaScript's 53-bit precision is crucial for system interoperability. This limitation necessitates parsing such identifiers as strings within JavaScript to avoid precision loss. For this purpose, libraries like json-bigint can be employed as an alternative to the native JSON parsing mechanism.

This approach facilitates the creation of identifiers across different systems without relying on a single point of failure, while also ensuring that the identifiers remain sortable. It underscores the importance of designing system architectures that can accommodate large-scale, distributed environments by employing robust data handling practices to maintain the integrity and order of identifiers across various platforms.

How to check

> **Note**
>
> In the current Watcher API, the creation of episodes is not implemented, so a temporary solution involves creating them through Central. This approach is planned to be changed in the near future.

To create episodes in Central, the following are required:

- The server address where Central is located.
- An API KEY, which can be obtained from `/etc/central/central.conf`.

If Central is running behind Flussonic, requests will be proxied, so there is no need to directly access port 9019 (the default port for Central).

curl \ -H 'Authorization: Bearer 08926CEA-3B54-4C77-99E2-A1F5FE54B11F' \ -H 'Content-Type: application-json' \ -d '{"episode_id": 1722279170848854016,"media":"cam0","opened_at":1699458327000,"updated_at":1699458327000}' https://watcher-server/central/api/v3/episodes/1722279170848854016

This call is sufficient to create a discrete event where the start and end times coincide.

To update it later, the same request can be made again but with updated fields:

curl \ -H 'Authorization: Bearer 08926CEA-3B54-4C77-99E2-A1F5FE54B11F' \ -H 'Content-Type: application-json' \ -d '{"episode_id": 1722279170848854016,"media":"cam0","opened_at":1699458327000,"updated_at":1699458338399,"preview":"..."}' https://watcher-server/central/api/v3/episodes/1722279170848854016

`preview` must be filled with actual `base64(jpeg)` of your picture.

Now you can see new episode in the Watcher.

3. Solutions

3.1 Advertising on Cameras

Advertising on cameras addresses the following tasks:

- Allows monetization of public cameras.
- Makes copying the stream from your site pointless.

You can monetize the camera by attracting clients to your site if you advertise your video surveillance service, or you can negotiate with an advertiser to show their clip on your service. A camera that shows ads is less attractive to webmasters who want to steal content from your site. In that case, their site will show your ads.

We have provided you with the ability to upload your clips and selectively show them on specific cameras. In this article, I'll explain how to do this.

3.1.1 Preparation

The player shows video clips as advertisements. You can't use images, PDFs, or HTML. We support mp4 files with the h264 codec. This is the most popular format, and preparing such a clip should not be a problem.

For example, I took the cartoon Big Buck Bunny, which is under a free license, and I will advertise Blender.

The following command downloads and trims the file to 10 seconds:

ffmpeg -i https://download.blender.org/peach/bigbuckbunny_movies/big_buck_bunny_480p_h264.mov -t 00:00:10 -c:v h264 -vf scale=-2:480 -c:a aac -f mp4 clip.mp4

I suggest you use any convenient video editor. Most often, the advertiser will help you with the mp4 file. Contact us if you need help preparing the content.

3.1.2 Uploading the Clip

- Open the **Admin** menu, go to the **Advertising** page.
- *(optional)* Insert the link to the advertised site. In my case, it is `https://studio.blender.org/films/big-buck-bunny/`.
- Upload the .mp4 file we prepared in the previous step.
- Click **Create**.

The added clip will appear in the list. You can check its functionality by clicking.

3.1.3 Creating a Share with Ads

- Open the **Admin** menu, go to the **Cameras** page.
- Select the desired camera.
- Go to the **Sharing** tab.
- Enter a name for the share. *I specify "for the landing page"*
- Check the **Advertising** box.
- Click **Create**.

After creating, the share will appear in the list and you will have two links available: - **Player live** - a link that can be copied to the clipboard. You can send this link in a chat or send it to the website developer; he knows how to embed it. - **HTML Live** - clicking here will copy the ready HTML code to the clipboard for embedding on the website.

3.1.4 Testing the Ads

To test the ads, use the **Player live** link. Copy and open it in a separate browser tab.

You should see the ad clip, and clicking on it will open the advertised site. After the clip plays, you will see the camera stream.

3.1.5 Embedding on the Website

We provide ready HTML code for embedding on the website. It is suitable for any website builder. If necessary, adjust the width and height parameters to fit the player into your site. Those who develop their own site may be interested in our player library.

3.1.6 Final words

Watcher allows you to upload video clips with ads and selectively enable them on specific cameras. You can influence development by leaving feedback through a ticket.

3.2 Embedding cameras on your website

A player with any camera from Watcher can be embedded into your external web pages such as a corporate website or a user's account.

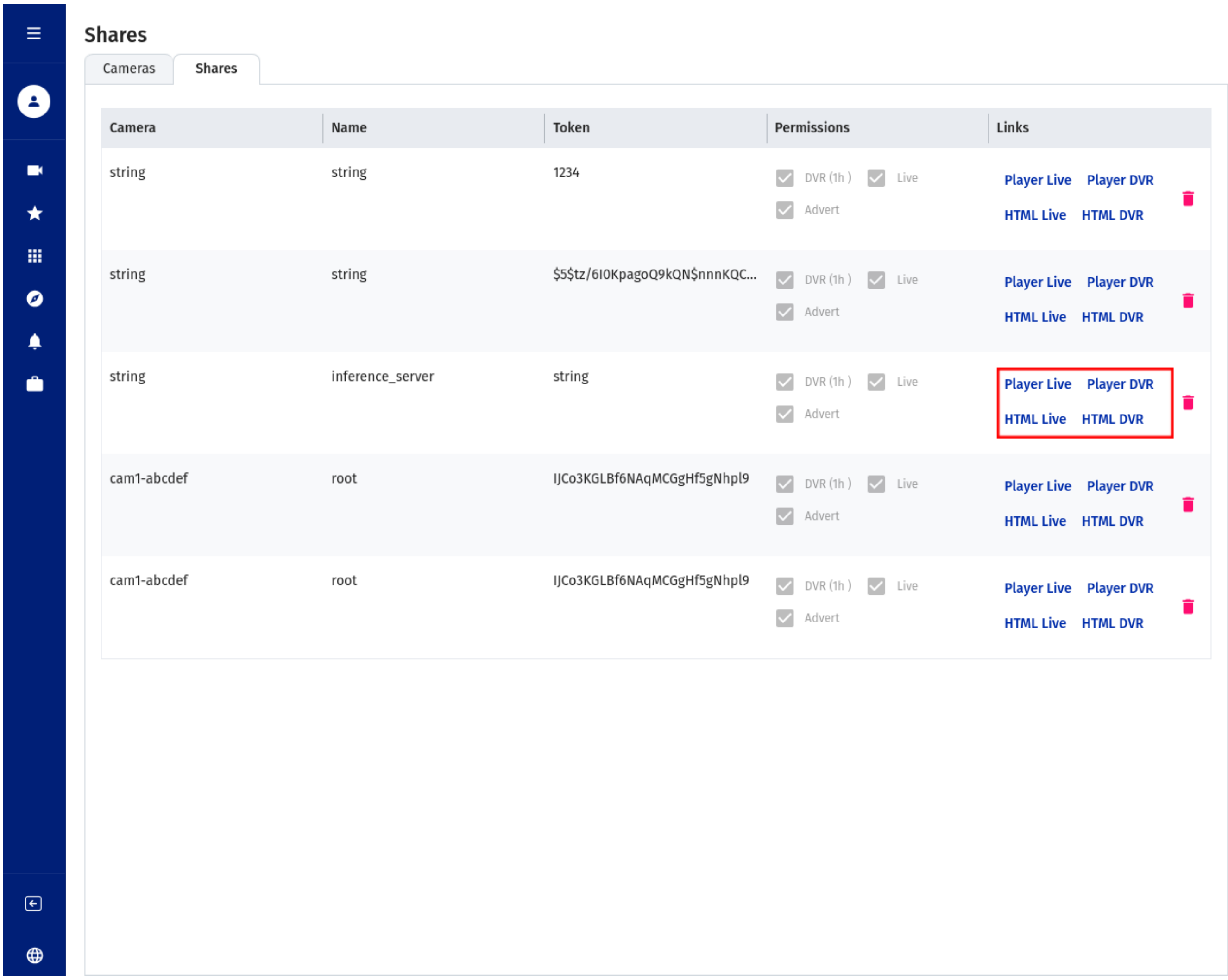
The method of embedding a camera described on this page allows the administrator to receive an URL to view any camera of any user. Authorization is carried out using a token. The administrator can add and remove tokens in the UI.

Regular users cannot yet share their cameras using the mechanism described here. They only have access to the old method of sharing from the Cameras page. We plan to upgrade the token management mechanism taking into account user rights.

3.2.1 Getting embed URL

To get the URL for embedding the player with the camera to your website:

1. Log into Watcher with admin permissions.
2. On the **Cameras** page (for admins), select the camera you want to embed.
3. Create an authorization token on the **Shares** tab.
4. Copy the link:
    - **Player Live** is for sharing live video only. Open this URL in a browser.
    - **Player DVR** is for sharing both live and archive. Open this URL in a browser.
    - **HTML Live** is an HTML code for embedding live video only. Paste the copied tag to the code of your website.
    - **HTML DVR** is an HTML code for embedding both live and DVR. Paste the copied tag to the code of your website.



Camera	Name	Token	Permissions	Links
string	string	1234	☑ DVR (1h) ☑ Live ☑ Advert	Player Live Player DVR HTML Live HTML DVR
string	string	$5$tz/6I0KpagoQ9kQN$nnnKQC...	☑ DVR (1h) ☑ Live ☑ Advert	Player Live Player DVR HTML Live HTML DVR
string	inference_server	string	☑ DVR (1h) ☑ Live ☑ Advert	Player Live Player DVR HTML Live HTML DVR
cam1-abcdef	root	IJCo3KGLBf6NAqMCGgHf5gNhpl9	☑ DVR (1h) ☑ Live ☑ Advert	Player Live Player DVR HTML Live HTML DVR
cam1-abcdef	root	IJCo3KGLBf6NAqMCGgHf5gNhpl9	☑ DVR (1h) ☑ Live ☑ Advert	Player Live Player DVR HTML Live HTML DVR

Note

If the embed URL does not work, check if you have specified the streamer's public payload URL correctly as instructed in Adding and configuring streamers

3.2.2 Terminating access to the camera

You can forbid access to the camera with any of the tokens at any time. Simply delete the token by clicking the trash can icon next to it. A camera embedded via an URL with a removed token will no longer be displayed on external websites.

3.2.3 Statistics of token usage

To explore statistics of camera views with specific token, use Retroview statistics service. Paste the token into the **User ID** field to display statistics on it.

3.3 Search the archive for visitors

When an incident occurs in a private facility, such as an office or workplace, the usual suspects are strangers. With Watcher, you can find videos of unregistered visitors and investigate the incident.

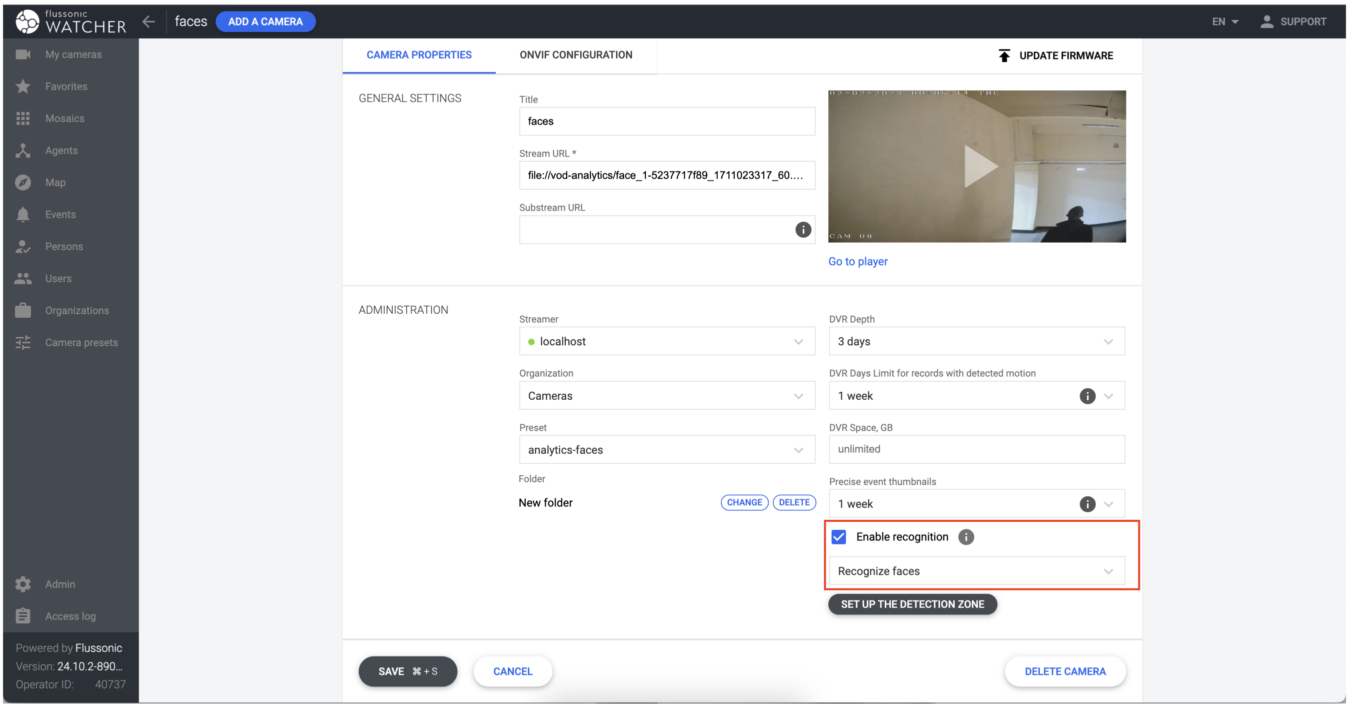
3.3.1 Set up the camera

Search for visitors is based on face recognition. So check that the camera is up to recommended video image parameters for face recognition.

3.3.2 Enable face recognition Watcher

To enable face recognition in Watcher:

1. Install the analytics module
2. Add the camera to Watcher.
3. Select the face recognition option on the camera.



GENERAL SETTINGS

Title: faces

Stream URL *: file://vod-analytics/face_1-5237717f89_1711023317_60....

Substream URL:

ADMINISTRATION

Streamer: localhost

Organization: Cameras

Preset: analytics-faces

Folder: New folder

DVR Depth: 3 days

DVR Days Limit for records with detected motion: 1 week

DVR Space, GB: unlimited

Precise event thumbnails: 1 week

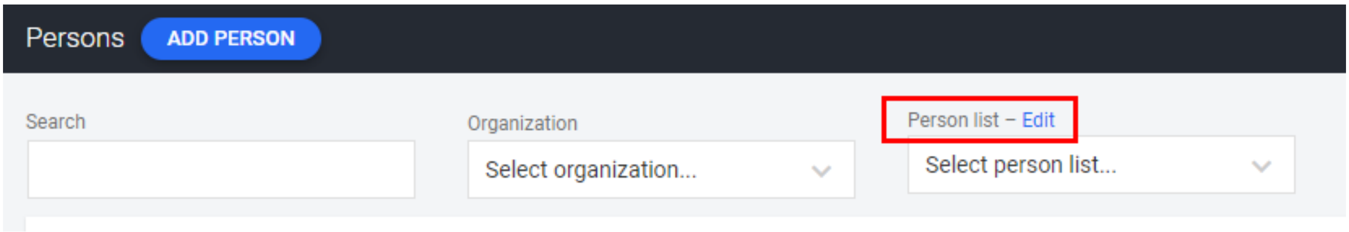
☑ Enable recognition

Recognize faces

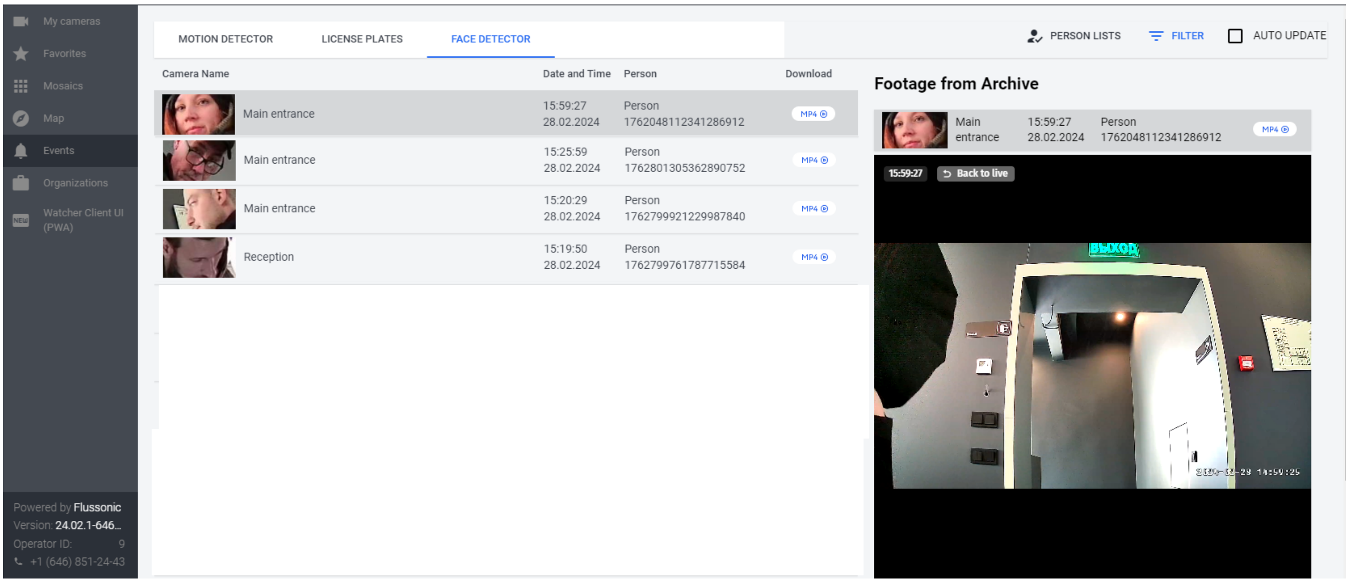
After that, Watcher will start detecting people's faces in the video and record these events in the camera's archive. You may find all the entries on the **Events** page in the Watcher UI.

3.3.3 Create accounts for the employees

1. Create a face list called "Employees". This can be done at **Events** - **Face Detector** - **Face Lists** - **Edit**.



2. Find a face among the faces detected by the camera and edit its profile by adding it to the "Employees" list. Everyone who is not on this list is considered a stranger.

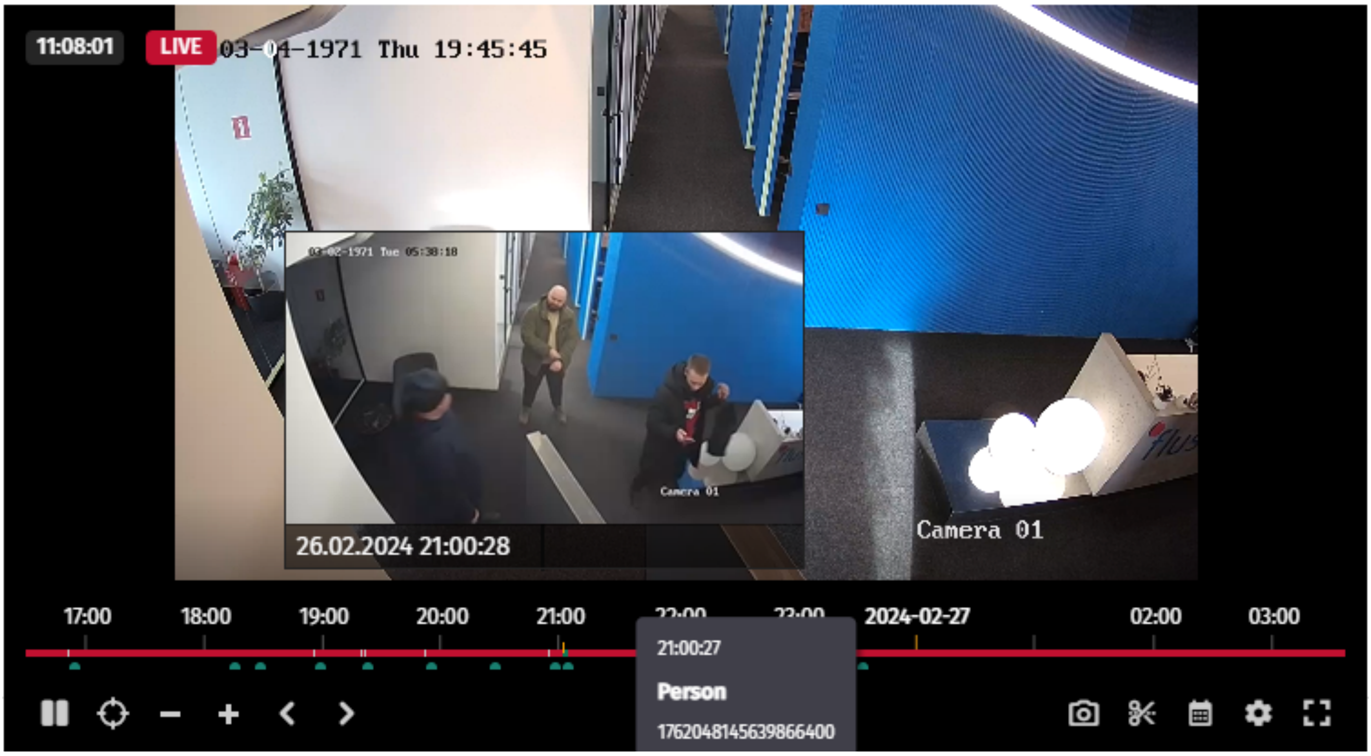


Camera Name	Date and Time	Person	Download
Main entrance	15:59:27 28.02.2024	Person 1762048112341286912	MP4
Main entrance	15:25:59 28.02.2024	Person 1762801305362890752	MP4
Main entrance	15:20:29 28.02.2024	Person 1762799921229987840	MP4
Reception	15:19:50 28.02.2024	Person 1762799761787715584	MP4

Footage from Archive: Main entrance, 15:59:27 28.02.2024, Person 1762048112341286912, MP4

3.3.4 Search for visitors on the video

Now you can see the marks on the time scale when watching video from your cameras. Persons added to the list are marked in green, and strangers are marked in red. Use the mouse wheel to focus on the area of the archive with a red mark. Click the mark to view the recording with the stranger.



3.4 How to Provide RTSP Access for an External System

Flussonic Watcher allows an external system to access video via the RTSP protocol, solving the following tasks:

- Playback of archives
- Uniform access from any streamer in the cluster to all cameras in the system (no need to change camera addresses in the external system)
- Access control and revocation of rights
- Reducing the load on the camera by multiplexing requests to it

3.4.1 How to Do It?

You must have Watcher installed. Whether it's an installation on a single server or in a cluster doesn't matter, the access method is the same.

Next, you need to create a camera to which you are going to provide access. To provide working access, it's necessary to create a long-term access token, a mechanism similar to adding a camera to a website.

Create a new token on the Cameras -> Shares page.

Then, you need to form the following RTSP URL:

```
rtsp://WATCHER-IP/CAMERA_NAME?token=TOKEN
```

Where you need to substitute the following values:

- `WATCHER-IP` - the address or hostname of the server with Watcher, on which Flussonic is running. It's important to note that you can use the address of any streamer in your cluster here, as they will all serve RTSP requests.
- `CAMERA_NAME` - the camera identifier. It's easiest to take it from the URL.
- `TOKEN` is taken from the table with tokens on the Shares page.

3.4.2 Uniform Access in the Cluster

Using Flussonic Central as an orchestrator for the Watcher cluster out of the box enables video playback from any camera on any streamer.

This will automatically enable restreaming of the stream between streamers.

This functionality is needed to avoid editing the list of sources in the external system every time a camera is moved.

You can specify either the address of the Flussonic that is installed alongside Watcher on the first server or any other streamer.

3.4.3 Playback of Archives

Access to the archive via RTSP will work automatically, without any settings.

It is only important that playback of the archive is allowed in the token itself.

3.4.4 Multiplexing Access

The approach with RTSP restreaming is needed for cases when there are many consumers for a camera: emergency services, internal affairs, healthcare, analytics, etc. The increased load will be on the streaming servers, but to the cameras themselves, there will be no more than one connection.

In this case, if the streamer is changed, the access settings for your partners will not change, which greatly facilitates working with them.

3.5 Viewing Watcher cameras on SMART TV, STB and IPTVPORTAL applications

You can enhance your package of services with the capability to view cameras on such subscriber devices as SMART TV, STB, and mobile applications of IPTVPORTAL.

On this page, we will tell you how to configure playback of video from Flussonic Watcher cameras through the **IPTVPORTAL** platform.

Before you configure broadcasting, you must have Flussonic Watcher installed, a camera with an archive must be added to it, and there must be an account in the IPTV Portal.

3.5.1 Watcher settings

When requesting video from set-top boxes, user authorization on the IPTVPORTAL must be used. To do this, open `/etc/watcher/watcher.conf` and add the custom_auth option specifying external authorization:

```
custom_auth http://DOMAIN.iptvportal.ru/auth/flussonic/arescrypt/;
```

Where DOMAIN is your domain in IPTVPORTAL; if Middleware is installed locally, use `http://SERVER_MIDDLEWARE/auth/flussonic/arescrypt/`.

3.5.2 IPTVPORTAL settings

Open the administrative panel of the IPTVPORTAL `https://HOSTNAME.admin.iptvportal.ru/` and add a new web camera (Media > Webcams). When adding a web camera, specify:

- **name**

  Enter the name of the camera that will be displayed in the IPTVPORTAL application.

- **mrl**

  Link to live broadcast from Watcher. The link should look as follows: `http://STREAMER/CAMERA_ID/mpegts`, where:

  - STREAMER - the hostname or IP address of the streamer to which the camera is connected
  - CAMERA_ID - the camera ID in Watcher

- **auth**

  flussonic arescrypt

- **timeshift url**

  URL for accessing the camera archive. Specify http://STREAMER/CAMERA_ID, where:

  - STREAMER - the hostname or IP address of the streamer to which the camera is connected
  - CAMERA_ID - the camera ID in Watcher

- **archive depth**

  Specify the number of days of the archive to be displayed in the IPTV Portal application, for example, `1 days, 0:00:00`.

- **timeshift auth**

  flussonic arescrypt

- **viewing devices**

  In the drop-down list, select those devices on which camera playback will be available:

  - rubrics. In the list the categories, choose where the camera will be added for search in the navigation menu.
  - packages. Select the service packages where this camera will be included.

Now create an IPTVPORTAL user account, assign a package to it, and include in that package the created camera.

Open the client part of IPTVPORTAL to see the live video and archive of this camera.

3.6 Using Zones to Distribute Cameras

Zones let you control which streamers a camera is assigned to, based on network or geographic location.

Note

The streamer cannot be changed directly. To configure the distribution of cameras across streamers, use zones.

⚠ **Warning**

The streamer cannot be specified directly because:

- During backup recovery, streamers with the old hostname will not exist.
- If streamer_hostname is specified, failover becomes unclear—yet it must be implemented.

3.6.1 Strict Distribution (required zone)

If a camera must run only on streamers within a specific network:

- Create a zone associated with this network.
- Assign streamers to this zone by navigating to the `Streamers` section and selecting the appropriate streamer(s).
- In the camera configuration, set this zone as `required`.

⚠ If there are no available streamers in the zone, the camera will not be assigned and won’t start.

3.6.2 Preferred Distribution (preferred zone)

If a camera should preferably run in a specific geo-region, but fallback is allowed:

- Create a zone associated with this network.
- Assign streamers to this zone via the `Streamers` section.
- In the camera configuration, set this zone as `preferred`.

The camera will be assigned to this zone if streamers are available. If streamers are unavailable, it will be assigned to any available streamer. Once streamers in the preferred zone become available again, the camera will move back to that zone automatically.

4. Administration Guides

4.1 Selecting hardware

4.1.1 Selecting hardware for video surveillance

When planning to install a video surveillance system at a secured facility or as a service for your subscribers, you will inevitably face the need to select servers or choose a plan on a cloud platform so that there were enough capacity to provide the required performance. You will have to understand what channel bandwidth you need, how much disk space, which CPU, how much video memory, etc. All these characteristics can be easily calculated if you know the number of cameras and the quality of the video coming from them, the number of subscribers/users, as well as the types of analytics that you plan to use.

You will find minimum hardware requirements on the System requirements for Watcher page. When using video analytics (face recognition and/or LPR), the requirements are slightly increased as described here. You can calculate the hardware characteristics for your case or run the load tests.

You will find the following guidelines in the sections below:

- Minimum system requirements
- Calculation of the bandwidth and disks for storing the video archive
- CPU selection
- Calculation of resources for analytics

4.1.2 System requirements for Watcher

Watcher mobile app and web UI have some software requirements as well.

On this page, you will find the following requirements:

- Single mode
- Cluster mode
- The Watcher mobile app
- Browsers

Single mode

Management + streaming server (two in one):

- *Operating system:* Ubuntu 22.04 LTS or later, Astra Linux Common Edition 2.12 or later, Astra Linux Special Edition 1.7 or later
- *Database:* PostgreSQL 14.4 or later;
- *CPU:* CPU: Xeon E-3 1230v5 3.4 GHz and higher;
- *Memory:* 32GB RAM;
- *Dedicated server:* Yes.
- *Hard drive type:* HDD / SSD;
- *Hard drive size:* depends on the camera network size and video archive storage requirements. See also Calculation of the bandwidth and disks for storing the video archive

```
10 GB of free disk space per 1 Mbps camera per day.
20 GB of free disk space per 2 Mbps camera per day.
70 GB of free disk space per 1 Mbps camera 7 days archive;
```

```
These system requirements are suitable for a network of:
500 1Mbps cameras
500 users
Failover turned off
Mosaic turned off
```

Or

```
250 2Mbps cameras
500 users
Failover turned off
Mosaic turned off
```

Servers should be dedicated, so no other software should run on them.

Cluster mode

Management server (the endpoint — it runs Flussonic Media Server, Flussonic Watcher, and the database):

- *Operating system:* Ubuntu 22.04 LTS or later, Astra Linux Common Edition 2.12 or later, Astra Linux Special Edition 1.7 or later
- *CPU:* CPU: 2-core CPU;
- *Memory:* 8Gb RAM;
- *Virtual server support:* Yes;
- *Database:* PostgreSQL 14.4 or later;
- *Hard drive type:* SSD;
- *Hard drive size:* 64GB of free disk space;
- *Dedicated server:* Yes.

Video streaming server (the streamer — it is used for video streaming and video archive storage):

- *Operating system:* Ubuntu 22.04 LTS or later, Astra Linux Common Edition 2.12 or later, Astra Linux Special Edition 1.7 or later
- *CPU:* CPU: Xeon E-3 1230v5 3.4 GHz and higher;
- *Memory:* 32GB RAM;
- *Dedicated server:* Yes.
- *Hard drive type:* HDD / SSD;
- *Hard drive size:* depends on the camera network size and video archive storage requirements. You can use the following data as a reference for calculations:
    - 10 GB of free disk space per 1 Mbps camera per day.
    - 20 GB of free disk space per 2 Mbps camera per day.
    - e.g. 70 GB of free disk space per 1 Mbps camera 7 days archive.

  See also Calculation of the bandwidth and disks for storing the video archive

Note

The above server requirements are given as an example suitable for 500 cameras x 1 Mbps, 500 users, failover enabled and mosaics not in use.

When increasing the bitrate to 2 Mbps, the number of cameras per 1 server should be reduced by 2 (down to 250). The recommendations are relevant only given that there will be no other running software on the servers.

Warning

For correct operation, Flussonic Watcher requires open ports 80, 443, 1935, 554 on all hosts, and the management server must have a real hostname that is resolved from the Internet.

The Watcher mobile app

Operating system:

- iOS 10 or higher
- Android 6 or higher

Browser requirements

Recommended for using Watcher:

- Mozilla Firefox 70 or higher
- Google Chrome 79 or higher

Not recommended (Watcher can function with some restrictions):

- Internet Explorer 11.356.18362.0 or higher
- Microsoft Edge 80 or higher
- Safari 13 or higher

4.1.3 Calculation of the bandwidth and disks for storing the video archive

The HDD space and network load are closely interrelated and depend primarily on the video quality, or bitrate for simplicity, and the number of cameras.

Video quality and bitrate are defined by the same parameters, therefore, it is more convenient to use bitrate as a numerical characteristic of quality when calculating hardware. However, you should always keep in mind that the quality itself does not have a numerical expression, is a subjective value only conditionally depending on the bitrate. For more information on what video quality is and how to find the balance between quality and bitrate, see our blog. By following this link, you can also find a few hints on how to calculate the bitrate if it is not shown in the camera specification or if you cannot to measure it experimentally.

Network load

The network load can be inbound and outbound. The inbound load depends on the number of cameras, and the outbound one depends on the number of simultaneous views.

To calculate the network load, it is enough to multiply the total bitrate by the number of cameras or views, respectively. However, you should anticipate the situation when all subscribers/users are simultaneously watching video, therefore, you should use the number of subscribers rather than the number of views when calculating the outbound load. We also recommend that you make about 30% margin of the channel bandwidth in order to provide for bitrate variability during the broadcast.

Inbound network load *= Number of cameras × Camera bitrate + 30%*

Outbound network load *= Number of users × Camera bitrate + 30%*

HDDs

The amount of disk space required for storing the archive is determined by the inbound network load, i.e. the number and bitrate of cameras as shown above, and the depth of recording storage. Usually, the archive depth is measured in days and depends on the area of application: a depth of up to several years may be required in the legal field, while a few days are usually sufficient for household cameras.

For calculation of the archive size, it is enough to multiply the inbound network load by the required archive depth. For example, one 1 Mbps camera with 30% margin is 1.3 Mbps; it takes about 14 GB of storage space per day; a 2 Mbps +30% = 2.6 Mbps stream will sum up into 27 GB per day, etc. So you will need at least 98 GB disk space to store a 7-day archive from a 1 Mbps camera, and if your system includes 500 such cameras, you will need 48 TB of disk space.

But you should not rush to immediately buy one dozens-of-terabytes HDD. When choosing the number and size of disks, you should also take into account some nuances:

- **The archive can occupy no more than 90% of the disk space**. This is due to file system peculiarities: it can slow down when the disk is almost full. To take this factor into account, the calculated archive size will need to be divided by 0.9.

  So, one 1 Mbps camera will require 16 GB per day instead of 14 as shown above. It makes into at least 52 TB for 500 such cameras with 7-day-deep archive.
- **The disk write and read speed limits the allowable inbound and outbound network load**. A large number of simultaneous read and write operations can reduce disk speed drastically. This is especially critical for a video surveillance service, but it can also affect the traditional video surveillance system: when dozens of cameras simultaneously write an archive to different areas of the disk, and dozens of users simultaneously view the archive, the disk speed can drop from the declared 100 MBps to 20-30 MBps (=160-240 Mbps).

  Simply put, the total of 1 Mbps cameras simultaneously writing to the disk and subscribers reading from the disk should not exceed 160-240, or 123-184 taking into account the 30% margin.
- Disk Requirements: 7200 rpm, SSD (for cache) + HDD for recording

Taking into account the above considerations, the HDD parameters are calculated as follows:

Disk space *= Inbound network load × Archive depth / 0,9*

Number of disks *= (Inbound network load + Outbound network load) / Disk speed*

***Disk size** = Disk space / Number of disks*

Example

Let's summarize the above calculations for the video surveillance service having the following parameters:

- 500 cameras with a bitrate of 1 Mbps
- 200 subscribers
- 7 days archive depth for all cameras (7 days × 24 hr/day × 60 min/hr × 60 sec/min = 604,800 seconds)
- Disk speed 20 MB/s = 160 Mbps

Inbound network load = 500 × 1 Mbps + 30% = 650 Mbps

Outbound network load = 200 × 1 Mbps + 30% = 260 Mbps

Disk space = 650 Mbps × 604,800 sec / 0,9 = 436,800,000 Mb ≈ 52 TB

Number of disks = (650 Mbps + 260 Mbps) / 160 Mbps ≈ 6 disks

Disk size = 52 TB / 6 = 9 TB

Thus, you will need to configure Flussonic RAID for DVR in the streamer with at least 6 HDDs of 9 TB each (or more disks of a smaller size if such a configuration is more appropriate economically).

CALCULATION OF THE NUMBER OF CAMERAS AND SUBSCRIBERS BASED ON DISK CHARACTERISTICS

If you have already purchased a server with a certain hardware configuration and you want to find out how many cameras and users can be connected to it, you can derive the following dependence from the given formulas:

***Number of cameras + Number of subscribers** = (Number of disks × Disk speed) / (1,3 × Bitrate)*

Let's try the reverse calculation for the example above:

Number of cameras + Number of subscribers = (6 × 160 Mbps) / (1,3 × 1 Mbps) ≈ 738

This is consistent with the source data (500 cameras + 200 subscribers), considering that we rounded the characteristics up.

> **Note**
>
> Remember that the camera number may be limited by the CPU performance, see CPU selection.

4.1.4 CPU selection

When calculating a platform for a video surveillance system or service, you need to know the maximum number of cameras that particular CPU can handle simultaneously. This may help avoid CPU overloading in production. However, since there is no universal officially recognized performance scale for all known CPUs, choosing generally comes down to simple brute force.

We've tested *Watcher* with a number of the most popular CPUs to provide general recommendations for choosing a CPU. The models given below as an example **are not** strictly recommended; these are just a few examples from our practice. The most accurate way of CPU selection remains to be load testing under the maximum load.

CPUs can be divided into low-, medium- and high-performance. The table below provides some examples and rough recommendations for the number of cameras.

Performance	Model example	Cameras per core
Low	Intel® Xeon® CPU X5675 @ 3.07GHz	up to 20
Middle	Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz	20 to 30
High	Intel(R) Xeon(R) CPU E3-1270 v3 @ 3.50GHz	more than 30

4.1.5 Calculating resources for analytics

Our face and license plate recognition modules are based on neural networks. We have created a separate neural network for each type of recognition consuming custom amount of video memory (VRAM) for GPU analytics or RAM for CPU. The analytics module memory consumption depends on the type of recognition you are going to to use.

Calculating video memory for analytics on GPU

You can use the following figures in your calculations of VRAM for analytics on GPU:

- License plate recognition consumes 1.7 GB of video memory per server and about 150 MB per each camera
- License plate recognition with detection of vehicles without license plates consumes 4.4 GB of video memory per server and about 150 MB per each camera
- Emergency vehicle detection consumes 4.4 GB of video memory per server and about 150 MB per each camera
- Face recognition consumes 2.5 GB of video memory per server and about 200 MB per each camera

It is also recommended to make a memory margin of about 1 GB for each neural network.

VRAM for one type of recognition = Video memory per neural network + Number of cameras × Video memory per camera + 1 GB

Please refer to this page for other system requirements for analytics on GPU.

Example:

Let's consider the situation when Watcher controls access to some facility. For that, there are 5 checkpoints each having one camera recognizing license plates to open the barriers, and 10 checkpoints each having one camera for face recognition to open the doors. Assume that we connect all these cameras to one streamer which will perform both license plate recognition and face recognition.

VRAM required for license plate recognition: 1,7 GB + 5 × 150 MM + 1 GB ≈ 3,4 GB

VRAM required for face recognition: 2,5 GB + 10 × 200 MB + 1 GB ≈ 5,4 GB

Total VRAM: 3,4 GB + 5,4 GB = 8,8 GB

Knowing this number and versions of Compute Capability supported by Watcher (6.1, 7.5 and 8.6) you can choose your video card(s) on the Nvidia website

Calculating RAM for analytics on CPU

You can use the following figures in your calculations of RAM for analytics on CPU:

- License plate recognition consumes 450 MB of RAM per server and about 25 MB per each camera
- Face recognition consumes 250 MB of RAM per server and about 30 MB per each camera

We also recommend about 30% margin in RAM to avoid system overload.

RAM for one type of recognition = RAM per neural network + Number of cameras × RAM per camera + 30%

Please refer to this page for other system requirements for analytics on CPU.

Example.

Let's calculate for the same example as above: 5 cameras with LPR and 10 for face recognition.

RAM required for license plate recognition: 450 MB + 5 × 25 MB + 30% = 757,5 MB

RAM required for face recognition: 250 MB + 10 × 30 MB + 30% = 715 MB

Total RAM: 757,5 MB + 715 MB = 1472,5 MB ≈ 1,4 GB

4.2 Installation

4.2.1 Installing Watcher Cluster or Single

Installation main steps

1. Install Flussonic Watcher on the managing server. Skip the steps marked "For cluster only" if you have just one server. PostgreSQL, Flussonic Central and Flussonic Media Server are installed together with Watcher and configured automatically.

   To create a Cluster, proceed with the following steps:

2. Install Flussonic Media Server on all streaming servers.
3. Add the streamers to the Watcher settings.

All the steps are described below.

You can also update or roll back Watcher as described in Updating Watcher or rolling back to previous version.

Installing Flussonic Watcher

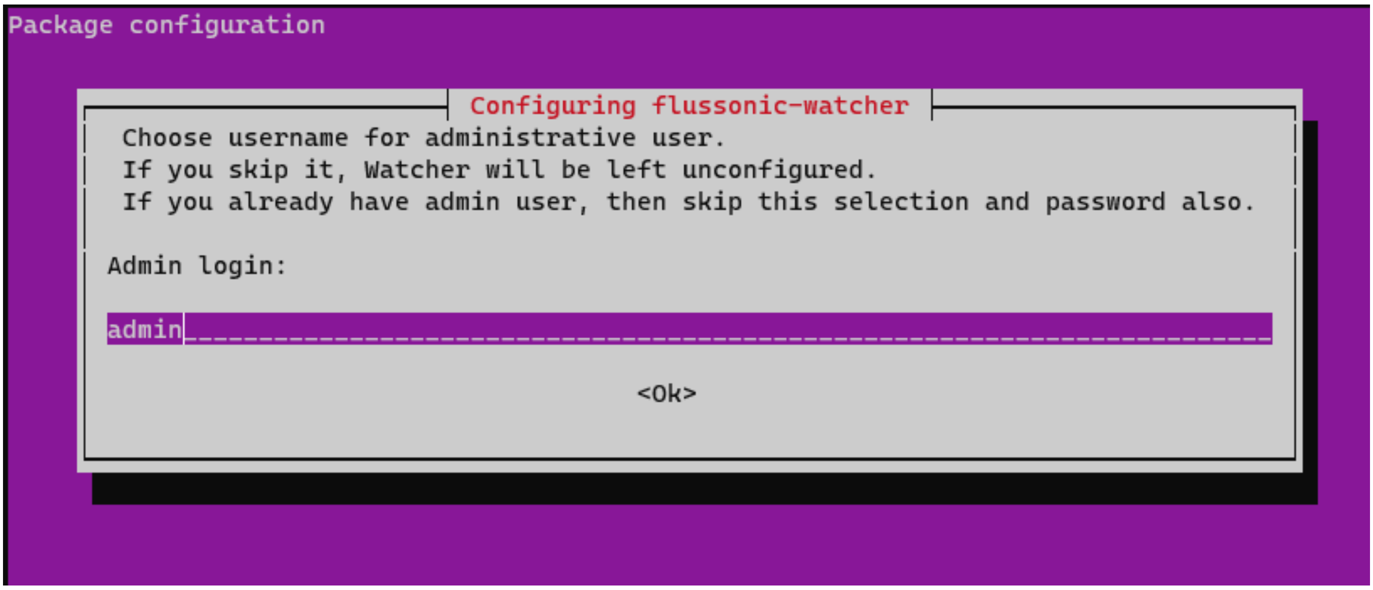
1) On the server where you plan to run Watcher execute the commands:

wget -q -O /etc/apt/trusted.gpg.d/flussonic.gpg http://apt.flussonic.com/binary/gpg.key; echo "deb http://apt.flussonic.com binary/" > /etc/apt/sources.list.d/flussonic.list; apt update;

Then use **one** of the following commands:

	apt install flussonic-watcher	apt install watcher-standalone
When to use it	* For Single * For Cluster, if the managing server shall act as a streamer too	* For Cluster only * The managing server shall only control the cluster; video will not go through it

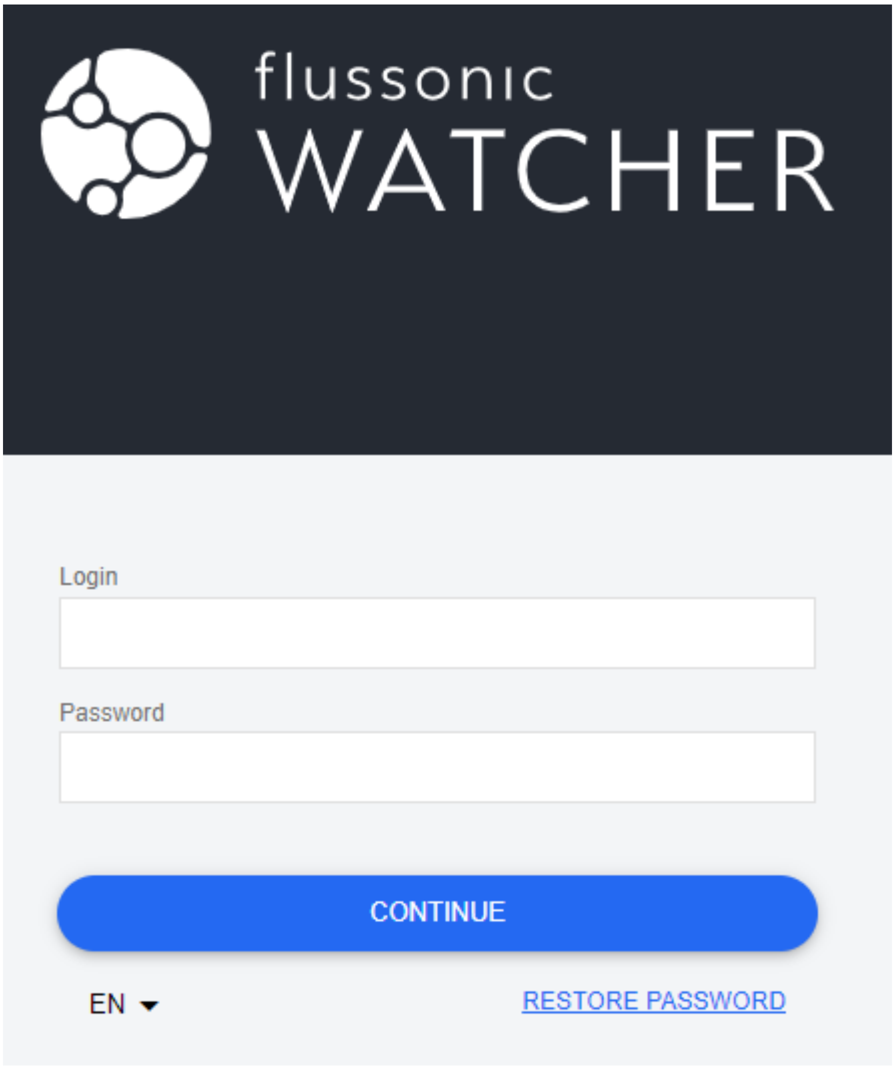
1) The installation wizard will prompt you to enter the necessary data: administrator's login and password, license key, and the path to the DVR where the archive will be stored.



```
Package configuration

 ┌──────────────────┤ Configuring flussonic-watcher ├──────────────────┐
 │  Choose username for administrative user.                           │
 │  If you skip it, Watcher will be left unconfigured.                 │
 │  If you already have admin user, then skip this selection and password also. │
 │                                                                     │
 │ Admin login:                                                        │
 │                                                                     │
 │ admin_______________________________________________________________ │
 │                                                                     │
 │                               <Ok>                                  │
 └─────────────────────────────────────────────────────────────────────┘
```

Enter the requested information and wait for the installation to complete.

3) Open Watcher web UI at `http://(Watcher server address)`. Use login and password you set at the installation.



The installation for single-server mode is now complete.

If you want to create a cluster, you will need to prepare streamers and set up Watcher to work as part of a cluster (see next steps).

Domain settings

For Watcher to work correctly, make sure to configure the DNS zone for the managing server. You need to add the A record in the DNS zone settings for your domain, and specify the hostname in it. This hostname must also be registered in the operating system on the managing server. This is necessary for streamers in you cluster, as well as Agents and mobile apps to access the server.

To check that the hostname resolves, run the `hostname -f` command on the server; it must return the correct hostname specified in the DNS settings, for example, example.com.

(For cluster only) Installing Flussonic Media Server on streamers

Flussonic Media Server shall be installed on all streaming servers.

1. Run the command:

   `curl -sSf https://flussonic.com/public/install.sh | sh`

2. Start Flussonic Media Server:

   `service flussonic start`

3. Go to `http://FLUSSONIC-IP/admin`, activate the license and set the following:

   a. On the **Config** page, change the administrator's login and password

   b. On the **Cluster** page, specify **cluster_key**: it must be the same as the cluster key for Flussonic Watcher; you can see it on the **Streamers** page in the profile of default `localhost` streamer created at installation.

4. Set the same date and time on the streamer as on managing server. For example, you can use the `timedatectl` utility for that.

(For cluster only) Creating a cluster (multiple server mode)

Creating a cluster means to add streamers (streaming servers) in the settings of Flussonic Watcher. **Streamers (streaming servers)** are servers intended to stream video from IP cameras. You must add at least one streamer on which IP cameras are added to configuration. This will allow you to start receiving video from cameras in cluster mode.

When the streaming server is ready for work, you need to add it in the settings of Flussonic Watcher:

1. Go to **Streamers**
2. Click **Create a streamer**
3. Enter the required settings:
   - **API URL** – the domain name of the streaming server.
   - **Cluster key** – the cluster key used in the cluster (the cluster_key option in the configuration file) that identifies a streamer belonging to a particular cluster.
   - Click **Issue by LetsEncrypt** to issue an SSL certificate for the server so that it can be accessed over HTTPS.

These and other settings are described in details on the Adding and configuring streamers page.

4.2.2 Updating Watcher or rolling back to previous version

We recommend updating Flussonic Watcher regularly to the latest release. Usually releases come out at the beginning of each month.

If necessary, you can always revert to a previously installed version.

You can also install a rolling update (the so-called master branch) of Flussonic Watcher on the Managing server and Flussonic Media Server on Streamers. The rolling update may have features that will be included in the next release but not yet available in the current release. This version is not guaranteed to be stable and should only be used for testing and evaluation purposes. Please use the release on live sites.

Updating or rolling back the Managing server

UPDATING WATCHER MANAGING SERVER

Run the following commands to update Watcher:

```
apt-get update
apt-get -y install flussonic-watcher
```

During the update, Watcher automatically migrates the database to work with the new version.

INSTALLING A ROLLING UPDATE (MASTER) TO THE WATCHER MANAGING SERVER

Remove the currently installed version of Flussonic Watcher and its dependencies:

```
apt remove flussonic-watcher
```

Change the repository to the one with the rolling updates and install Flussonic Watcher:

```
echo "deb http://apt.flussonic.com/repo master/" > /etc/apt/sources.list.d/flussonic.list;
apt update;
apt install flussonic-watcher;
```

RETURNING WATCHER MANAGING SERVER TO THE MAJOR RELEASE

Remove the currently installed version of Flussonic Watcher and its dependencies.

```
apt remove flussonic-watcher
```

Change the repository to the one with official releases and install Flussonic Watcher:

```
echo "deb http://apt.flussonic.com binary/" > /etc/apt/sources.list.d/flussonic.list;
apt update;
apt install flussonic-watcher;
```

Note

We strongly recommend that you back up your database every day and before you update Watcher.

ROLLING BACK WATCHER MANAGING SERVER TO THE PREVIOUS VERSION

To rollback to the previous version of Watcher, follow these steps:

- Create a backup copy of the database so that if necessary you can quickly restore the service.
- Determine dependencies:

  ```
  apt-cache show flussonic-watcher=20.06 | egrep 'Depends|Suggests:'
  Depends: flussonic (>= 19.12), flussonic-python (=20.05.1), postgresql (>= 9.6)
  ```

- Install the required version and its corresponding dependencies:

  ```
  apt install flussonic-python=20.05.1 flussonic-watcher=20.06
  ```

- Restart the service:

  ```
  service flussonic restart
  ```

Updating and rolling back the Streamers

UPDATING YOUR STREAMERS TO THE CURRENT RELEASE

Run these commands on each streamer:

apt-get update apt-get -y install flussonic service flussonic restart

INSTALLING A ROLLING UPDATE (MASTER) ON STREAMER

Remove the currently installed version of *Flussonic* and its dependencies:

apt remove flussonic

Change the repository to the one with rolling updates and install *Flussonic*:

echo "deb http://apt.flussonic.com/repo master/" > /etc/apt/sources.list.d/flussonic.list; apt update; apt install flussonic; service flussonic restart

RETURNING THE STREAMER TO THE CURRENT RELEASE

Remove the currently installed version of *Flussonic* and its dependencies.

> ⚠ **Warning**
>
> Before removing the packages, create a backup of the configuration files located in the directory `/etc/flussonic` and the `.db` files in the directory `/opt/flussonic/priv`.

apt remove flussonic

Change the repository to the one with official releases and install *Flussonic*:

echo "deb http://apt.flussonic.com binary/" > /etc/apt/sources.list.d/flussonic.list; apt update; apt install flussonic; service flussonic restart

> ⚠ **Warning**
>
> If *Flussonic* fails to start, run the commands `systemctl status flussonic.service` and `journalctl -xe` and send their output to our technical support team.

ROLLING THE STREAMER BACK TO THE PREVIOUS VERSION

To do this, you must specify the exact version of the `flussonic` package and its dependencies.

Suppose you want to revert to version 22.11.

Get dependencies' versions by using `apt-cache`:

apt-cache show flussonic=22.11 | egrep '^(Depends|Suggests):'

Result will be like:

Depends: flussonic-erlang (=24.0.6.3), flussonic-transcoder-base (=22.08.3)

Install packages with these versions:

apt-get install flussonic=22.11 flussonic-erlang=24.0.6.3 flussonic-transcoder-base=22.08.3

⚠ **Warning**

Before installing packages create a backup of the configuration files in the directory `/etc/flussonic` and `.db` files in the directory `/opt/flussonic/priv`.

4.2.3 Migrating Flussonic Watcher to a new server

Below is the procedure for the Flussonic Watcher managing server migration in Cluster. If you want to upgrade your Single installation to Cluster, i.e. split streamer and managing server by migrating your managing server to a separate server, please contact our technical support team at support@flussonic.com for help.

To migrate the managing Watcher server, follow these steps:

1. Update your existing server with Flussonic Watcher to the latest release.
2. Install Flussonic Watcher on the new server in a usual way (you may skip the last step of cluster install, i.e. conf file editing).
3. Copy and transfer the files `/etc/flussonic/flussonic.conf` and `/etc/flussonic/license.txt` to the new server.
4. On the old server, back up the database.
5. Transfer the resulting .sql file to the new server.
6. Restore the data from this file on the new server.
7. Restart the service:

   ```
   service flussonic restart
   ```

4.2.4 Database Backup in Watcher

Create a database backup:

- Regularly according to the schedule. In case of an unexpected failure, you can quickly restore the system.
- To migrate the Watcher database to a new server. Make a backup on the old server and deploy it on the new one.
- Before making significant changes to the configuration. If the change causes a service failure, you can roll it back.
- Before updating Watcher. If something goes wrong during the update, the data in the backup will remain intact.

Creating a Backup

You can create a database backup using the `watcher` utility:

```
/opt/flussonic/contrib/watcher backup create
```

After executing the command, a message will appear on the screen:

```
Create backup watcher database, please waiting...
Backup success to file /var/lib/flussonic-watcher/backups/watcher_dump_20240711_100517.bak.gz
```

⚠ **Caution**

It is highly recommended to set up regular automatic database backups.

Restoring from a Backup

To restore from a backup, run the command:

```
/opt/flussonic/contrib/watcher backup restore
```

By default, the utility will take the latest backup. Confirm by entering `y`.

```
Using the latest dump: watcher_dump_20240711_101404.bak.gz
Or specify a specific dump. Usage: /opt/flussonic/contrib/watcher backup restore watcher_dump_20240711_101404.bak.gz
File creation date: 2024-07-11 10:14:04
Are you ready to restore the dump? During this time, the services will be stopped (y/n)
```

Example of Scheduling Regular Backups

The simplest way to set up regular backups is to use the `cron` tool. Execute the following in the command line:

```
crontab -e
```

On the first run, the system will prompt you to choose an editor to open the cron configuration file. We recommend using `nano`, which is first in the list:

```
Choose 1-4 [1]: 1
```

Add a line at the end of the file, for example:

```
0 5 * * * /opt/flussonic/contrib/watcher backup create
```

This rule will backup the database every day at 5:00 AM. To set a different frequency, edit the parameters at the beginning of the command. For more details, see here.

After editing, press `Ctrl+O` to save, and then `Ctrl+X` to exit the editor.

Make a Copy to an External Server

It is good practice to store backups on a separate server, otherwise backups are not protected from:

- System disk failure
- File system corruption

We do not offer a ready-made solution for synchronizing files with an external disk. Choose a solution that will be most convenient in your case. If you need help, our support service will consult you.

4.3 Managing cameras and permissions

4.3.1 Presets management

A presets in Watcher is a set of DVR and analytics parameters that you can use as a template when creating and configuring cameras. When you select a preset in the camera settings, the parameters from the preset are populated to the camera settings. A set of presets on the camera is defined by the set of presets selected for the camera's Organization.

The presets can be adjustable or non-adjustable. A non-adjustable preset will not allow any user to change the populated settings manually, and an adjustable preset allows that.

Purposes of the presets:

1. If you are deploying a subscriber service, **presets correspond to billing plans**. In most cases, the non-adjustable presets are utilized in the subscriber services to forbid subscribers with permissions to edit cameras from changing DVR and analytics settings since these parameters are usually determined by the plan. Create and configure the presets before or in the same time with the billing system integration to provide your subscribers with flexible and affordable plans.
2. If you are arranging CCTV on a secure site, presets facilitate DVR and analytics settings for a bunch of cameras. Typically, adjustable presets are used for video surveillance systems in order to be able to individually change the parameters on each camera as needed.

User permissions required to manage or use presets:

- Watcher Administrator permissions to create, delete or change presets.
- Permission to edit Organizations to add presets to Organizations.
- Permission to edit cameras within an Organization to assign a preset to the camera within the Organization.

Any camera must have a preset selected for it. When you have just installed Watcher, there is a Default preset in it. The Default preset is available in all organizations.

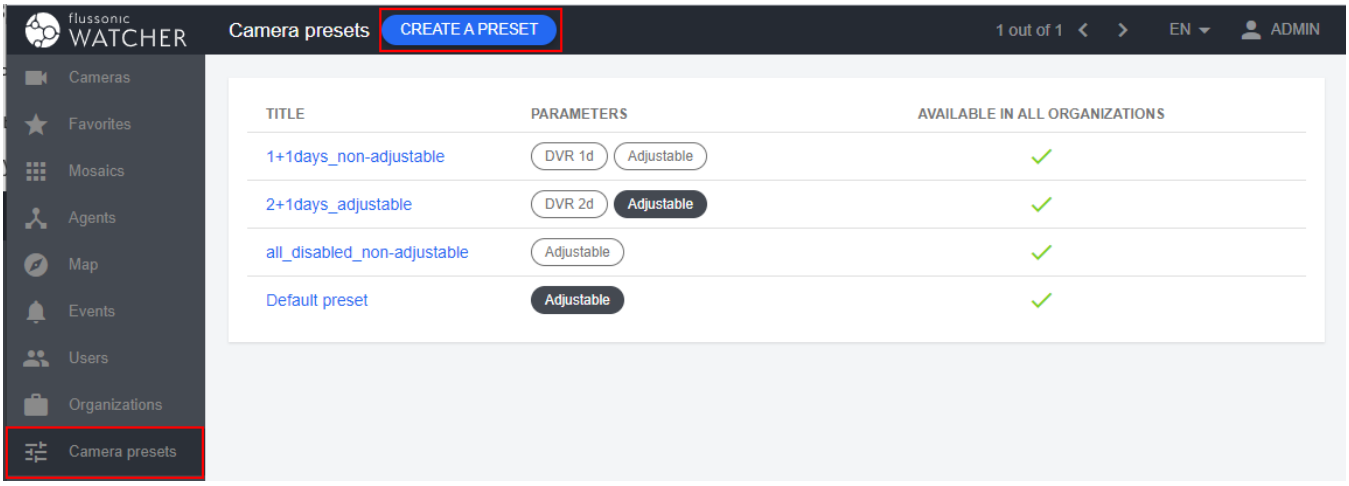
> **Note**
>
> If you are using Watcher as a CCTV system, the Default preset would be enough. However, feel free to use presets as you like to facilitate DVR settings for many cameras.

Creating a preset

To create a preset:

1. Select **Camera presets** in the main menu on the left of the page.

   The list of presets opens:

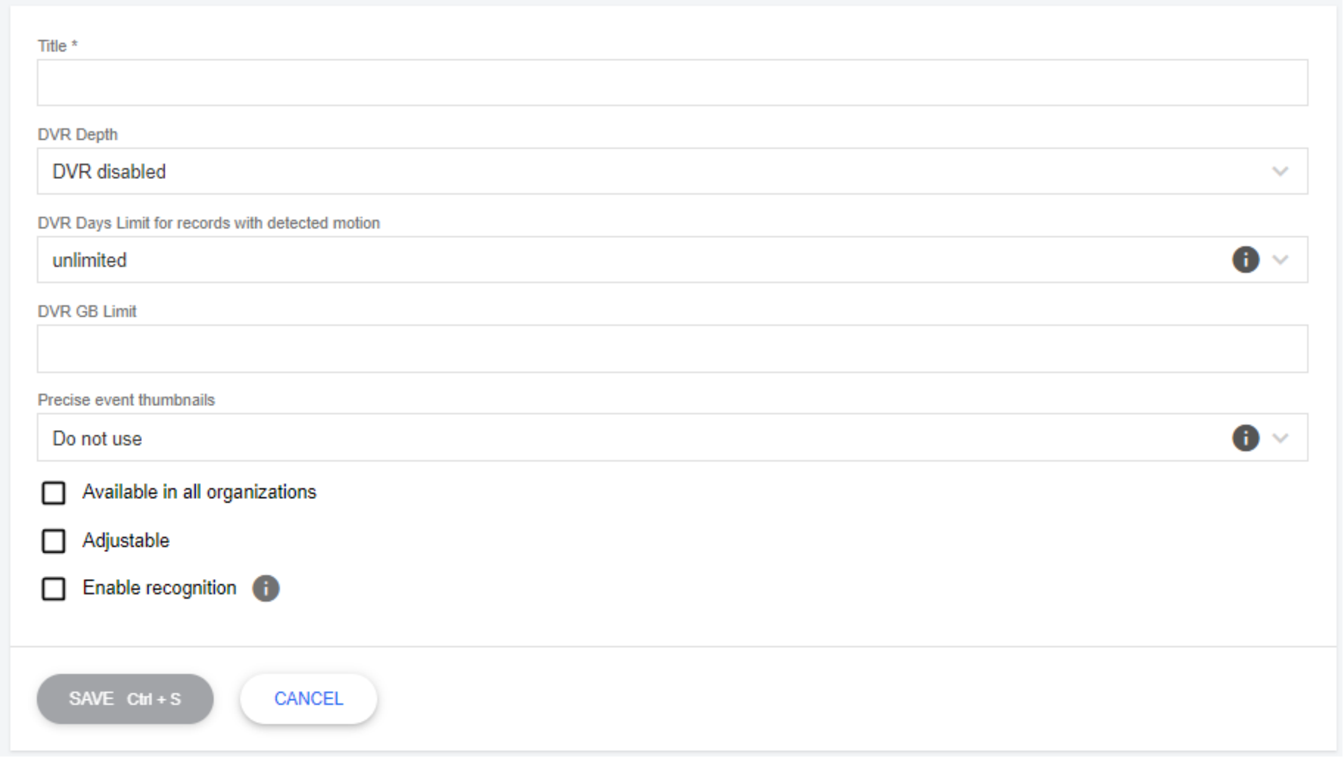


TITLE	PARAMETERS	AVAILABLE IN ALL ORGANIZATIONS
1+1days_non-adjustable	DVR 1d Adjustable	✓
2+1days_adjustable	DVR 2d Adjustable	✓
all_disabled_non-adjustable	Adjustable	✓
Default preset	Adjustable	✓

   You may select an existing preset to edit or delete some presets here. This page also shows a non-editable preset summary: DVR parameters, adjustability and availability in all Organizations. These parameters are set via the form described below.

2. Click **Create preset**.

   The form with preset settings opens:



Title *

DVR Depth
DVR disabled

DVR Days Limit for records with detected motion
unlimited

DVR GB Limit

Precise event thumbnails
Do not use

□ Available in all organizations

□ Adjustable

□ Enable recognition

> **Note**
>
> When you hover over the info mark, a pop-up box with the parameter explanation opens.

3. Change the following preset parameters as needed:

    a. **Title** is the name of the preset.

    b. **DVR depth**, **DVR days limit for records with detected motion**, **DVR space**, **Precise event thumbnails** are the archive parameters. Learn more at Archive settings.

    c. **Available in all organizations**: check the box to allow selection of the preset for any camera in any organization (i.e. any subscriber), or uncheck it to make the preset available only for the organization(s) where it is selected (see Adding organizations).

    d. **Adjustable**: if checked, allows changing the DVR settings for each camera with such preset individually; if unchecked, the DVR settings become disabled for the camera where such preset is selected.

    e. **Enable recognition**: if checked, one of the analytics type will be available on the camera with such preset: select license plates recognition (LPR) or face recognition in the list under the box.

After adding a preset you can select one or more of the created presets in the organization settings so that the presets become available in the camera settings (unless the preset is available in all organizations, then just select it for some cameras).

Please note that these selections should be made in accordance with the billing system integration: in some cases the billing system can forward all the settings to Watcher while in other cases only cameras are added using the billing system as a proxy with all presets, organizations, users settings made in Watcher.

4.3.2 Managing Organizations

Organizations in Watcher are logical structures uniting cameras and users by some criteria.

An Organization can be your subscriber's own space in Watcher where they can add cameras and give other users access to them: in this case you will need as many Organizations as you have subscribers. Or you can add housing estate cameras to the Organization and allow the residents to view cameras they need.

In addition, the cameras within the Organization can be grouped to folders with separate user permissions to each folder. In the mentioned housing estate example, you can group cameras installed outdoors and at entrance halls and provide the newly connected subscribers with access to the folders with outdoor and their entrance hall cameras.

Basically, Organizations and folders allow mapping the structure of your subscriber network in Watcher for easy further management.

If you use Flussonic Watcher as a CCTV on site, then a single Organization will be enough for you - it is created in Watcher by default.

To manage Organizations, go to **Organizations** in the Watcher main menu.

This page tells you how to:

- Add an Organization
- Edit an Organization
- Manage cameras within Organization

See also how to add users to an Organization and create mosaics .

Adding Organizations

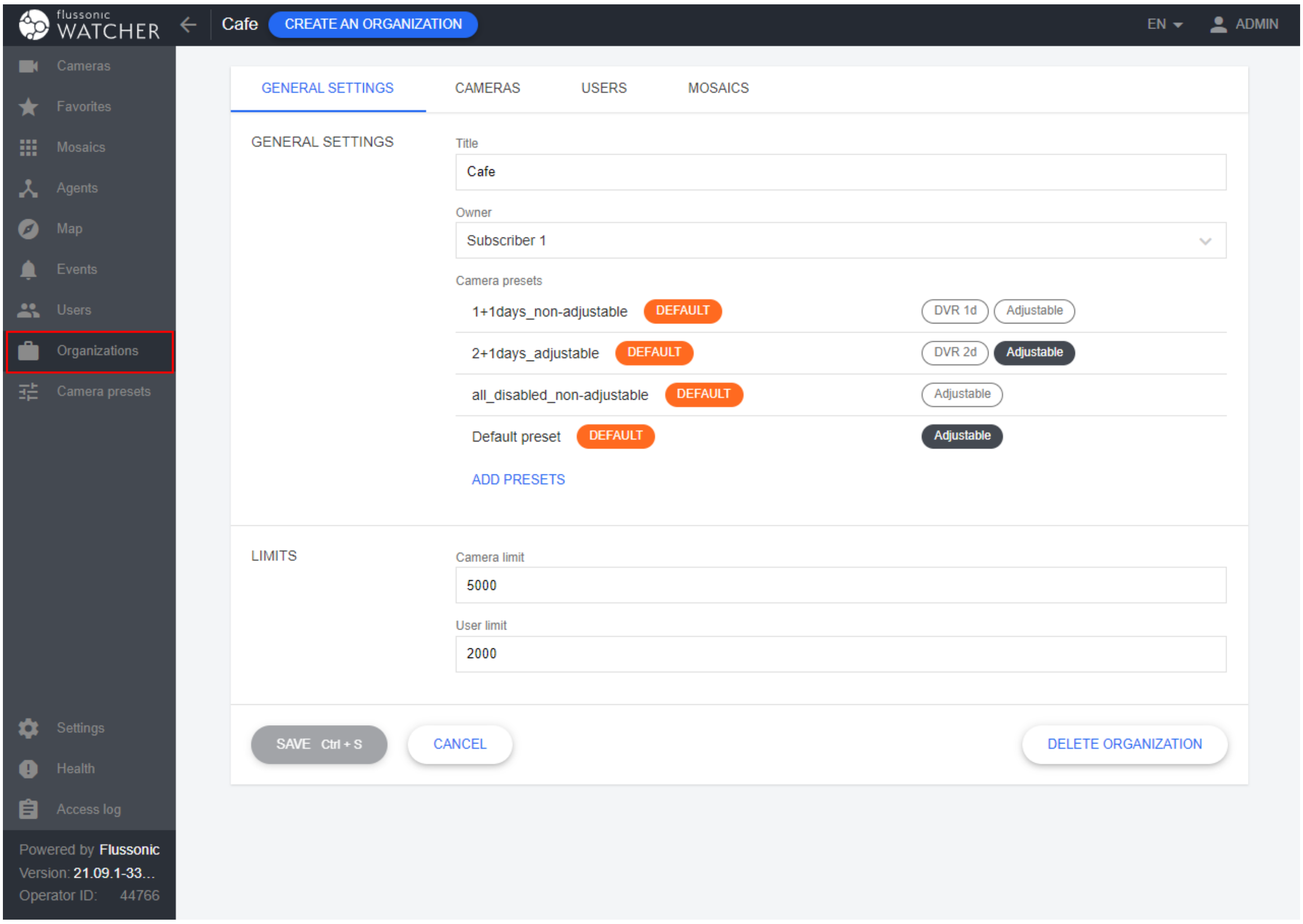
If you use Flussonic Watcher as a CCTV system on site, then you will be good with a single Organization created by default.

If you use Watcher to provide VSaaS, you need to create an Organization that corresponds to each subscriber.

> **Note**
>
> Before you add an Organization, you can create a user who will be the owner for this Organization and assign this user with the full set of permissions for managing the Organization. However, this is not necessary: you can create such user while creating an organization.

To add an Organization for a new subscriber:

1. Go to **Organizations** > **Create an Organization**
2. Fill in the general settings and click **Save**.



GENERAL SETTINGS

Title: Cafe

Owner: Subscriber 1

Camera presets

1+1days_non-adjustable	DEFAULT	DVR 1d, Adjustable
2+1days_adjustable	DEFAULT	DVR 2d, Adjustable
all_disabled_non-adjustable	DEFAULT	Adjustable
Default preset	DEFAULT	Adjustable

LIMITS

Camera limit: 5000

User limit: 2000

- **Title** - defines the name with which the Organization will be displayed in the list of Organizations.
- **Owner** - a Watcher user who has the maximum permissions for managing the Organization (meaning managing cameras, users, and settings) and who is responsible for payments for your service.

> **Note**
>
> The Watcher user to be assigned as an Organization owner must be created in Watcher *before* you create an Organization. If it is not created, you can create and add the user on the **USERS** tab.

- **ADD PRESETS** - click to select previously created preset(s) that should be available for the cameras within organization.
- **Camera limit** - the maximum number of cameras that can be added to a single Organization.
- **User limit** - the maximum number of users that can be added to a single Organization.

Having filled the general settings, you can add users, cameras and mosaics.

Editing Organizations

To edit an Organization, go to **Organizations** and click the title of the Organization you want to modify.

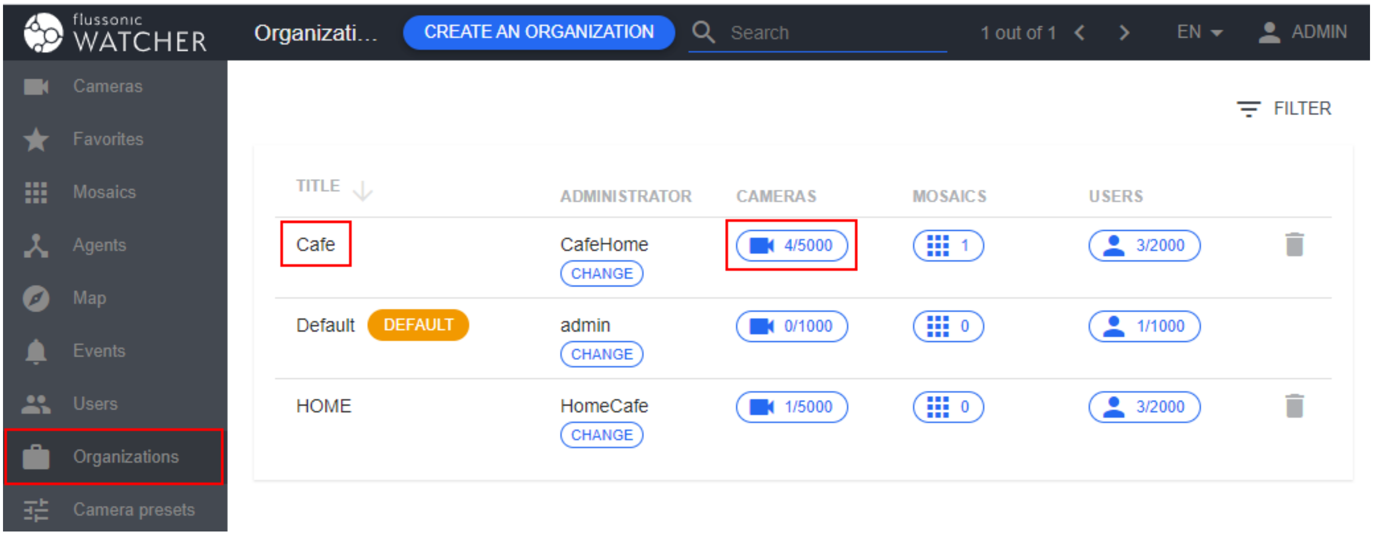
Managing cameras within Organization

The key element of each Organization is the list of IP cameras. One camera can belong only to one Organization.

You should select the **Organization** and (optionally) **Folder** when you add a camera or edit camera settings to add a camera to the Organization.

By default, Watcher uses an Organization marked as Default, and the root folder inside the default Organization.

If you need to view the list of cameras within Organization, go to **Organizations** and click the counter in the **Cameras** column next to the Organization name.

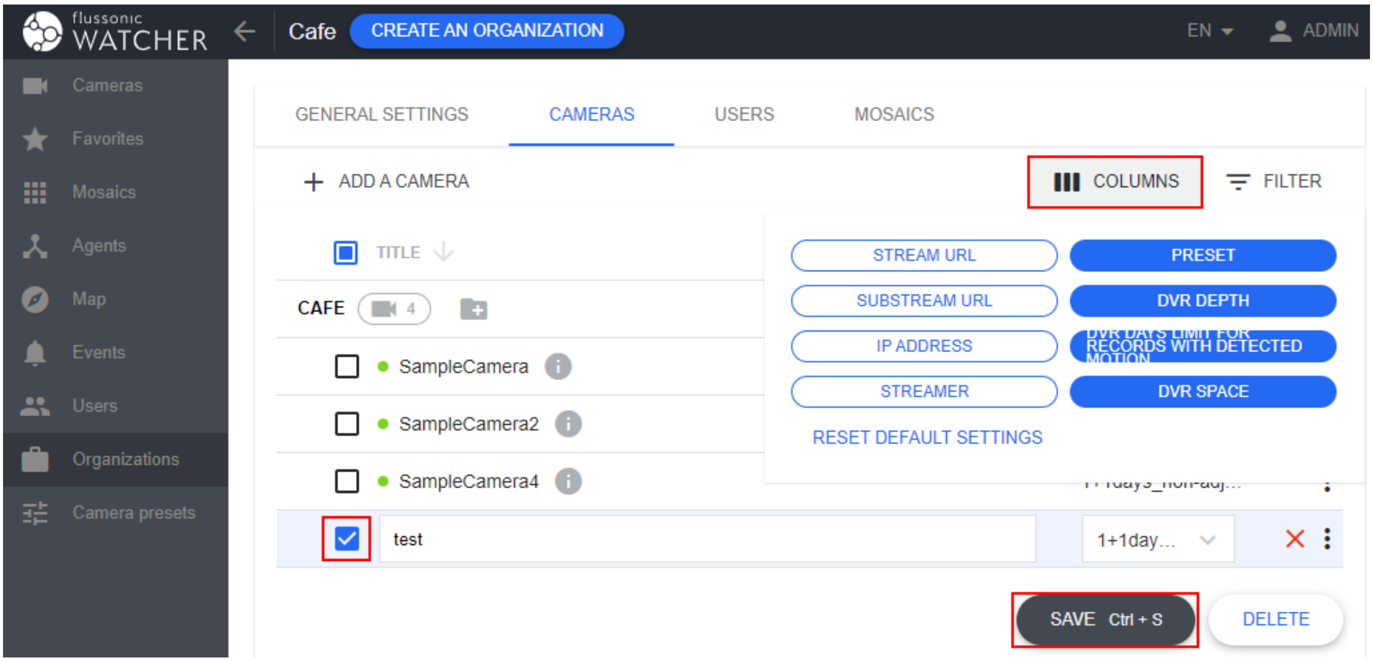


TITLE	ADMINISTRATOR	CAMERAS	MOSAICS	USERS
Cafe	CafeHome CHANGE	4/5000	1	3/2000
Default DEFAULT	admin CHANGE	0/1000	0	1/1000
HOME	HomeCafe CHANGE	1/5000	0	3/2000

Note

You can also select the **Cameras** tab while editing the Organization.

The **Cameras** tab opens. Here you can change camera settings or delete cameras from Watcher. Set the box next to camera name to enable the controls. You can also select displayed settings by the **Columns** button.



Click **Add a camera** to create a new camera. The camera settings page opens with Organization already selected. You should enter the rest of the settings.

4.3.3 Adding cameras

Supported cameras

Any camera with RTSP support can be added to Watcher.

However, access to cameras behind NAT always presents many troubles. This is why Flussonic introduces versatile tools to facilitate this process:

- Agent creates a secure tunnel between the camera and the server to which the Agent is bound.
- Iris firmware not only has a built-in Agent for plug-and-play connection with the server but also makes it easier to connect your camera to the Wi-Fi network. Moreover, Iris enhances the camera features in Watcher with motion detection and PTZ control (given the hardware support).

The additional features are also available for cameras with ONVIF support including motion detection, PTZ control, automatic network search and configuration of the camera itself from the Watcher interface.

Supported codecs

Watcher supports **H.264 and H.265 (HEVC)** codecs.

Remember that the use of H.265 (HEVC) depends on whether it is supported in a web browser. Most of the browsers including Safari, Chrome, and Edge, support H.265. However, H.265 playback may be unavailable in some browsers like Firefox.

Summary of adding cameras

Cameras in Watcher are created and exist only within a specific Organization. One camera can belong to one Organization only. Please refer to Managing Organizations section for details.

Depending on the scenario of using Watcher, cameras can be added by users with different roles including video surveillance system administrator, provider (or subscriber service) administrator, subscriber, etc. The user must have permissions to manage cameras in the Organization to which they add camera(s).

The cameras can be added via web UI or Watcher mobile app. Available ways to add a camera depend on the camera type.

> ⚠ **Warning**
>
> The camera with bound Agent is automatically provisioned to bound Watcher as soon as the camera connects to a network with Internet access. In some cases, the billing integration may be used for adding the camera to the proper Organization.

Ways to add cameras via Watcher mobile app:

Menu item \ Camera type	Camera with Iris	Camera with Agent	Common camera
Wi-Fi camera (via QR code)			
Ethernet camera (MDNS request)			
ONVIF camera			

Ways to add cameras via Watcher web UI:

Menu item \ Camera type	Camera with Iris	Camera with Agent	Common camera
Add a camera (manually by RTSP URL)			
Add by IP			(ONVIF only)
Search cameras			(ONVIF only)

You will find the detailed instructions on each way below on this page.

Note

If you have a large list of cameras with connection parameters in a CSV file, you can import this data to Watcher using API or web UI. Watcher administrator permissions are required for that.

Feel free to contact our technical support team in case if you haven't found a suitable way to connect your cameras. We are always ready to help you find a solution that suits your needs.

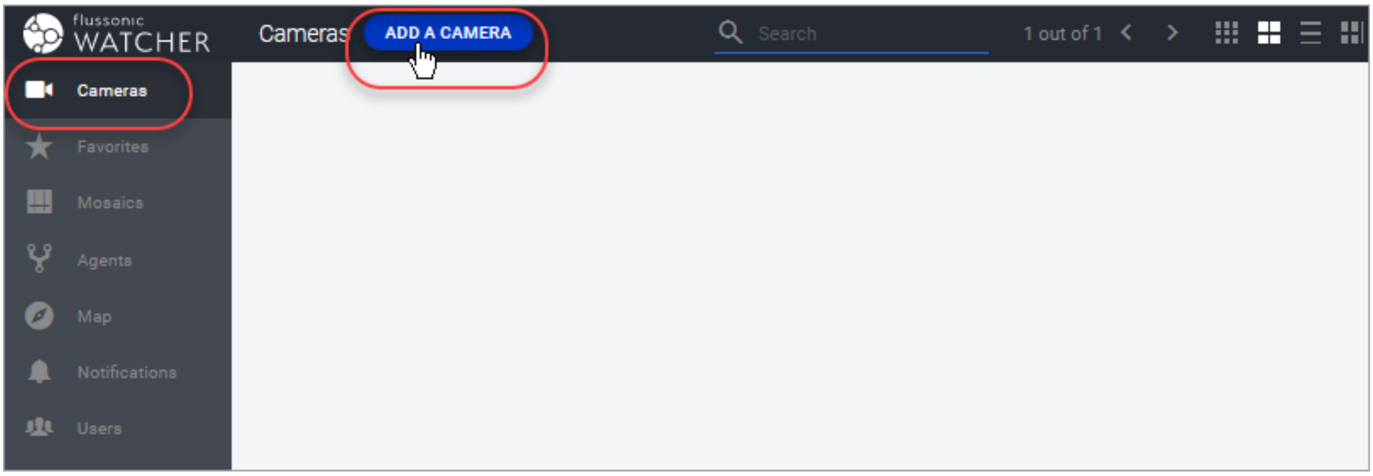
Adding cameras via web UI

ADDING CAMERAS MANUALLY

Any camera with RTSP support can be added manually. Please note that manual adding of cameras with Iris or Agent eliminates the benefits of these tools related to the tunnel for connection through NAT.

You should know the RTSP URL of camera streams in order to be able to add the camera manually. Please look for this info in the camera manual.

To add a camera by RTSP link, go to **Cameras** in the main menu and click **Add a camera**:



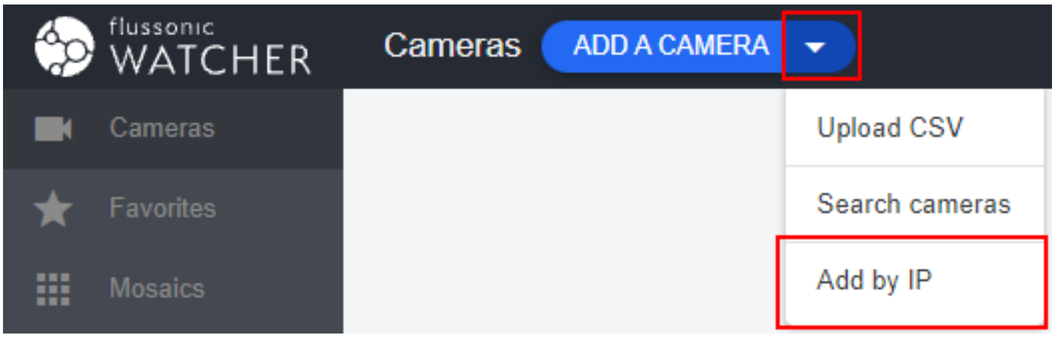
The camera settings form opens. Set the desired parameters.

ADDING CAMERAS BY IP

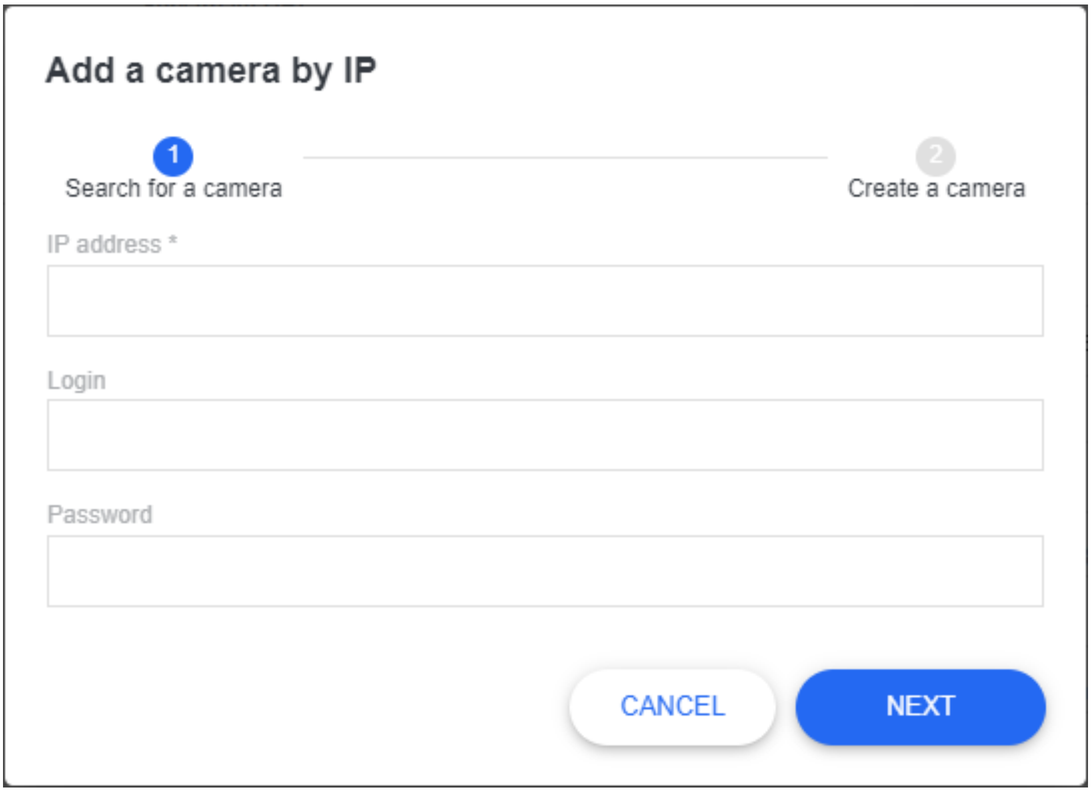
The camera must support ONVIF or have the Iris firmware in order to be added by IP. In addition, the camera should be in the same LAN with the Watcher managing server or streamer to which it is going to be added.

To add a camera by IP:

1. Go to **Cameras** and select **Add by IP**.

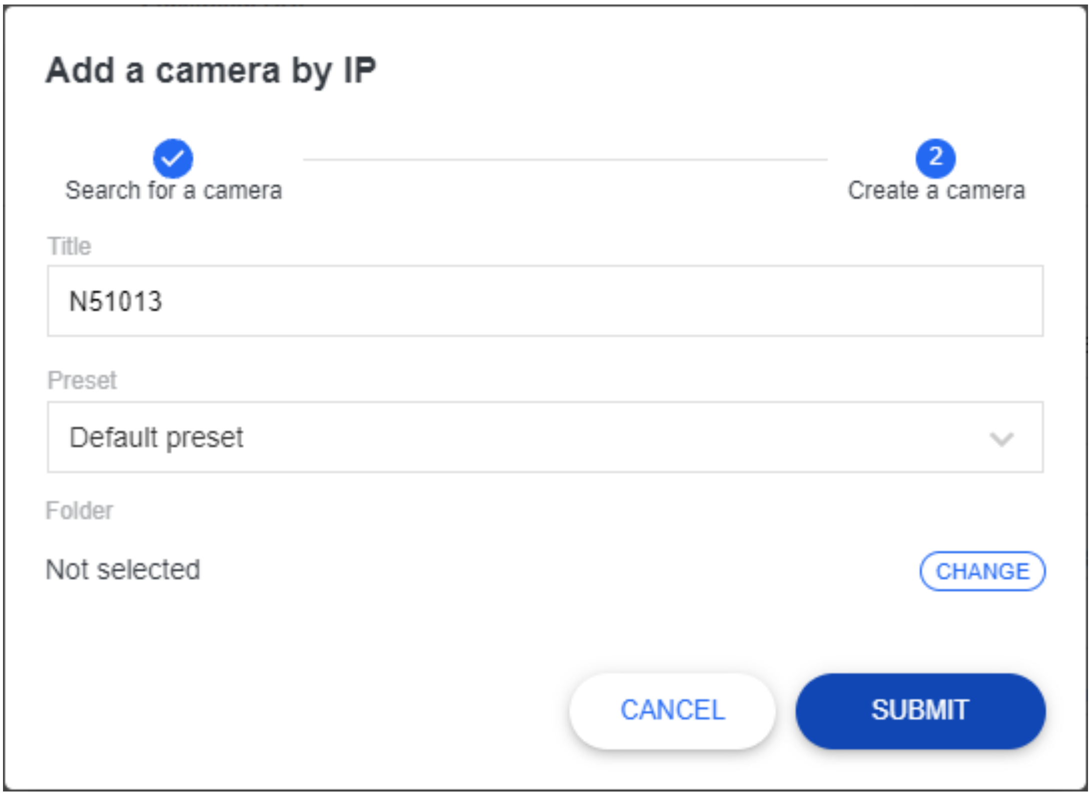


2. In the dialog that opens, type the camera IP address and credentials for connection (if necessary).



3. Click **Next** to check if the camera is available.

4. If Watcher successfully connects to the camera, a form will open for entering the name of the created camera, selecting a preset and a folder.



Add a camera by IP

Title: N51013

Preset: Default preset

Folder: Not selected

5. Having the parameters set, click **Submit**.

The added camera settings page will open. Configure and save the desired parameters if necessary.

SEARCHING FOR CAMERAS

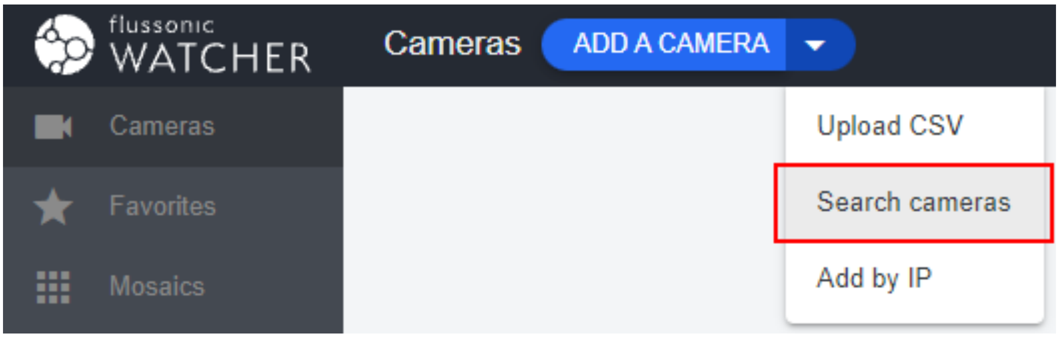
Flussonic Watcher uses the WS-Discovery mechanism to search ONVIF-compatible cameras. It can detect Ubiquity, Samsung and other cameras.

The user must have Watcher administrator permissions to be able to use this way of adding cameras.

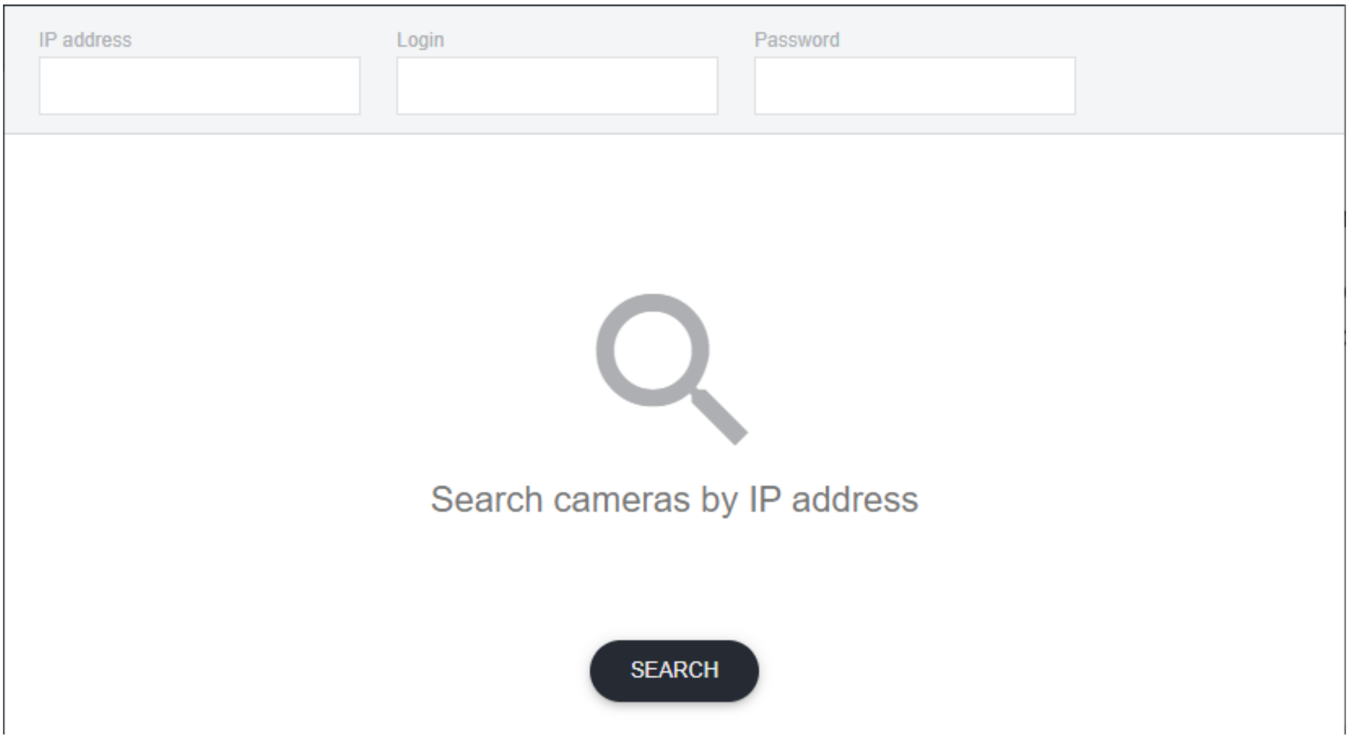
In this way, cameras can be searched only in the network of the Watcher managing server. If streamer(s) and Watcher for some reason are on different local networks, then it is recommended to add cameras by IP or manually instead of searching them.

To search for cameras:

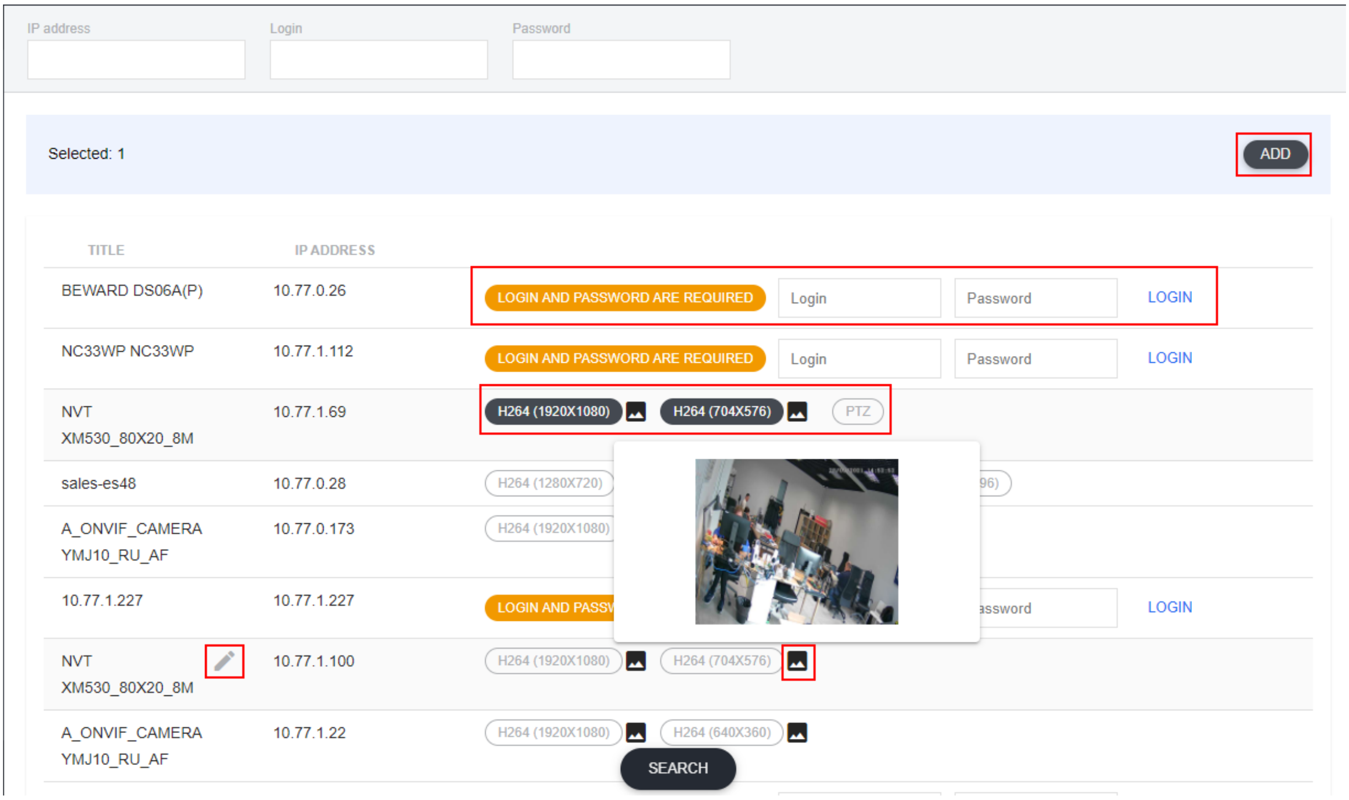
1. Go to **Cameras** and select **Search cameras**.



2. Click **Search** on the page that opens and wait for the search to finish.



3. As the search proceeds, the found cameras will appear on the page.



IP address	Login	Password

Selected: 1

TITLE	IP ADDRESS				
BEWARD DS06A(P)	10.77.0.26	LOGIN AND PASSWORD ARE REQUIRED	Login	Password	LOGIN
NC33WP NC33WP	10.77.1.112	LOGIN AND PASSWORD ARE REQUIRED	Login	Password	LOGIN
NVT XM530_80X20_8M	10.77.1.69	H264 (1920X1080)	H264 (704X576)	PTZ	
sales-es48	10.77.0.28	H264 (1280X720)	96)		
A_ONVIF_CAMERA YMJ10_RU_AF	10.77.0.173	H264 (1920X1080)			
10.77.1.227	10.77.1.227	LOGIN AND PASSW		assword	LOGIN
NVT XM530_80X20_8M	10.77.1.100	H264 (1920X1080)	H264 (704X576)		
A_ONVIF_CAMERA YMJ10_RU_AF	10.77.1.22	H264 (1920X1080)	H264 (640X360)		

Some cameras require login and password for authorization so you may have to enter your login and password to connect the camera.

Click the pencil icon that appears when you hover over the camera line in order to edit the camera name.

Many cameras has two or more H.264 streams, so you can add any or all of them.

When you hover the mouse cursor over the thumbnail icon, a preview of the stream from the camera pops up.

Also you can turn on PTZ control in the list of found cameras if PTZ is available.

4. Having the stream(s) selected, click **Add** in the top of the list.

The cameras will be added to Watcher with default preset to the default Organization. You will find them on the **Cameras** page and will be able to adjust the rest of the settings if necessary.

> **Note**
>
> If you cannot find the camera you need on the list but you know the camera's IP address and credentials, try using the form in the top of the page. If Watcher successfully connects to the camera at the specified IP, then you will be able to add it from the search page. Alternatively, you can add your camera by IP in this case.

Adding cameras via the Watcher mobile app

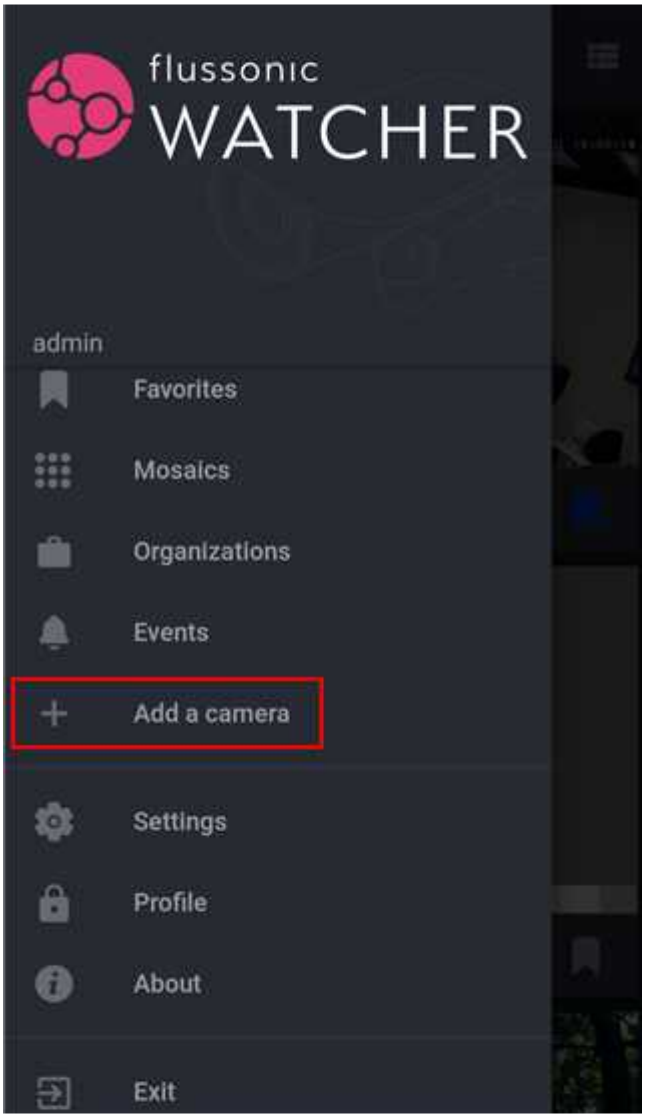
> **Note**
>
> The screenshots in this section were made with Android app. The iOS app appearance may differ, however the main steps are the same.

Before adding a camera via the Watcher mobile application, make sure that:

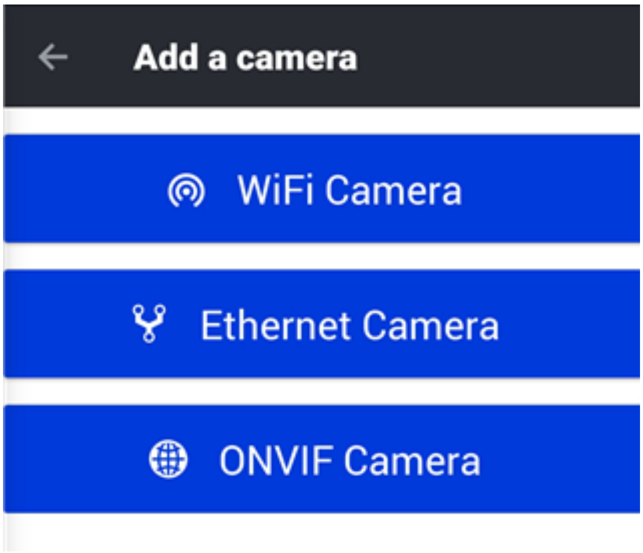
- The Flussonic Watcher mobile app is installed on the phone
- The user logs in to a mobile application using the credentials that grant the rights to edit the list of cameras in at least one Organization.

To add a camera via the mobile app:

1. Log in to the Flussonic Watcher mobile application using the credentials you received from the system administrator.
2. Open the menu and select **Add Camera**.



3. Tap the desired way to add the camera.



The ways are described below.

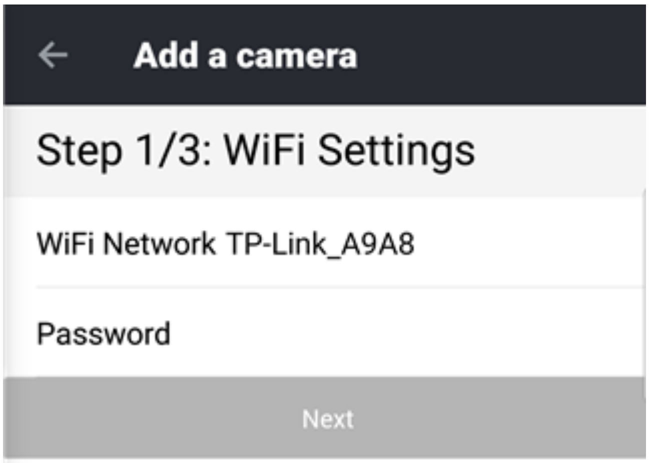
ADDING A WI-FI CAMERA WITH IRIS

Wi-Fi cameras with Iris firmware are supplied inactivated i.e. with Agent not bound to any Watcher server. In this case the Agent can be activated using the Watcher mobile application. During the activation process, the camera will connect to the Wi-Fi network and receive a command from the Watcher application to connect to an Organization on the server.

Make sure that the camera is reset to factory default settings before proceeding to the activation.

To add a camera with Iris via the mobile application:

1. Select **Wi-Fi camera** in the add camera menu.
2. Check the network name and enter the Wi-Fi password.



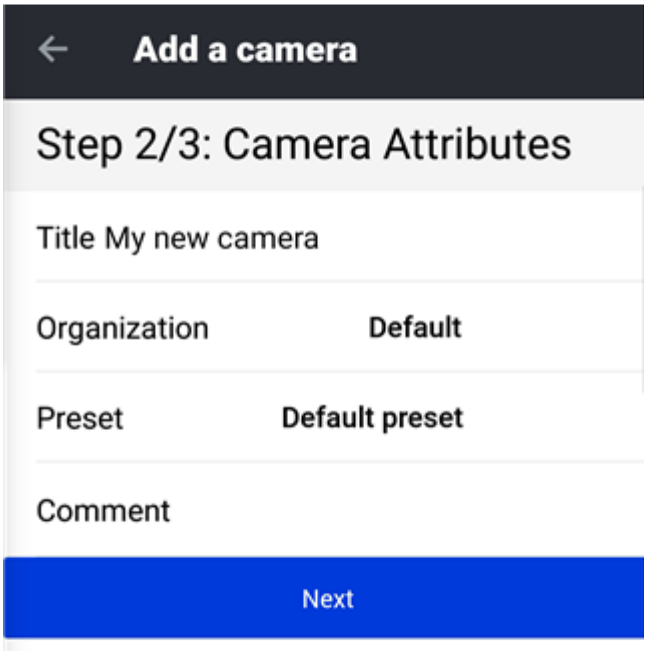
Add a camera

Step 1/3: WiFi Settings

WiFi Network TP-Link_A9A8

Password

3. Fill in the camera information:
   - Name is the name under which the camera will be displayed in Watcher.
   - Organization is the Organization in which you add the camera.
   - Preset is a group of recording settings to be set on the camera being added.
   - Note is additional information about the camera (not required).



Add a camera

Step 2/3: Camera Attributes

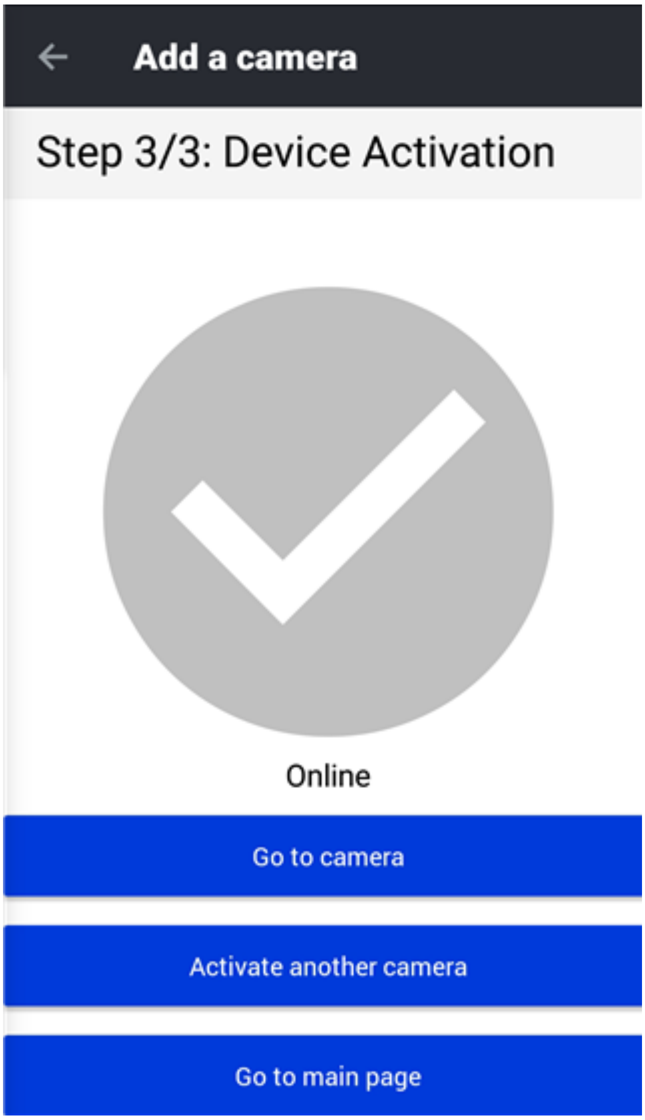
Title My new camera

Organization Default

Preset Default preset

Comment

4. Click **Next**. The QR code will be displayed on the screen.
5. Present the QR-code to the activated camera. Please note that if the camera is ready to read the QR code, it will play the corresponding voice message from time to time. If you do not hear the voice message, check the power connection and try resetting the camera to factory settings.
6. The camera will inform you when it has successfully read the QR code and connected to the Wi-Fi network. After that,, you can go to viewing the video from the camera, activate another camera, or return to the main page in the application.



Add a camera

Step 3/3: Device Activation

Online

Note

It may take some time for a newly connected camera to start sending frames. In practice, this time does not exceed one minute.

ADDING AN ETHERNET CAMERA WITH AGENT

If you connect cameras with unbound Agent (with or without Iris firmware) *via Ethernet*, they can be found on the LAN automatically or added by IP from the Watcher app. The search is carried out using the MDNS protocol, by sending a UDP request to the address 224.0.0.251.

Note

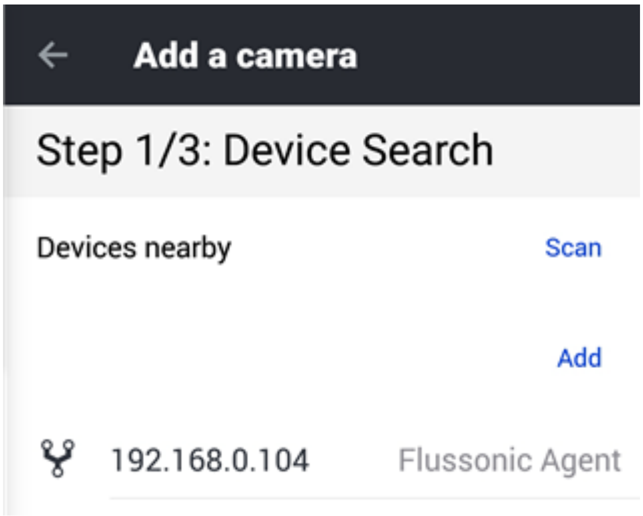
Make sure to configure your network so that MDNS requests were not blocked, e.g. disable IGMP snooping on your router.

Cameras must be connected to the same local network as the phone with the Flussonic Watcher mobile application installed.

The Agent will de activated when you add the camera this way.

To activate an Agent on an Ethernet camera:

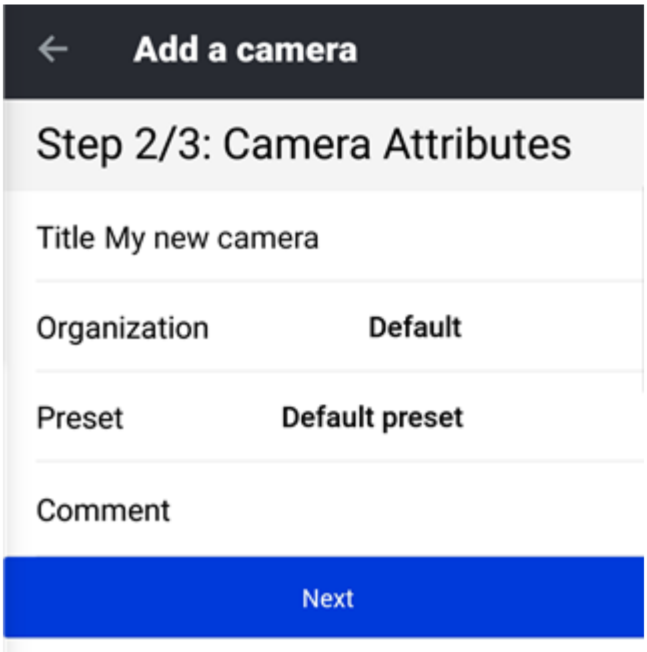
1. Select **Ethernet camera** in the add camera menu. The mobile application will scan the local network and you will see a list of found cameras.

   192.168.0.104 Flussonic Agent

   To start the search again, tap **Scan**. If the desired camera is not found, but you know its IP address, click **Add** and enter it.

2. Fill in the camera information (see item 3 in the above section for details):

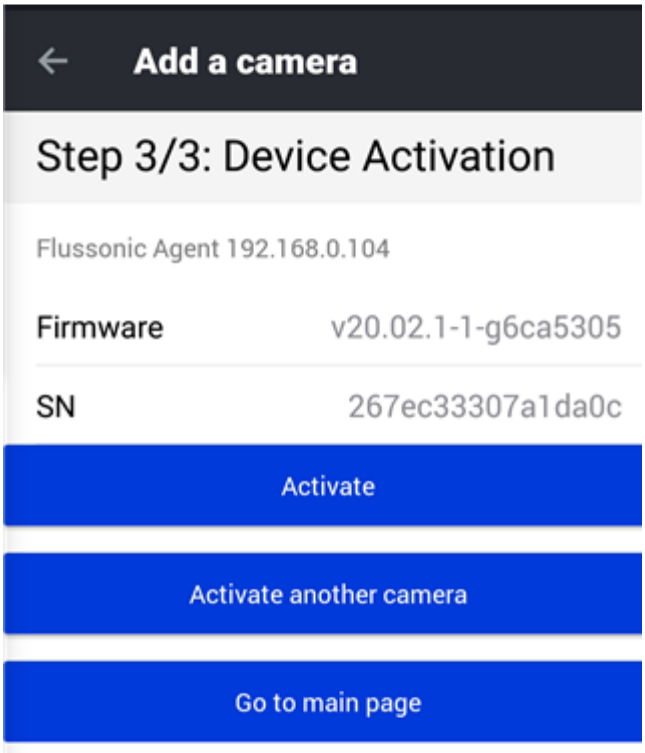
   Title My new camera

   Organization Default

   Preset Default preset

   Comment

3. Tap **Activate** on the next screen.



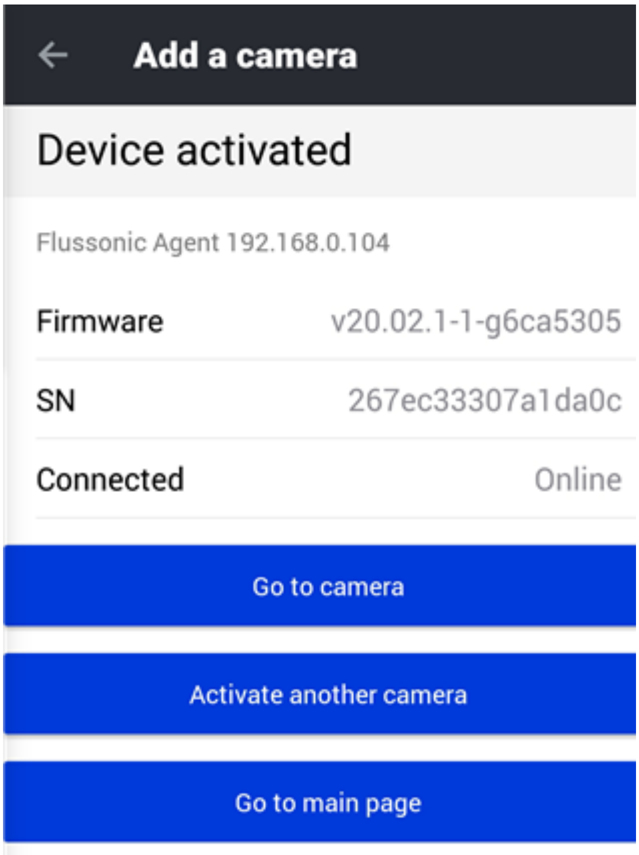
Add a camera

Step 3/3: Device Activation

Flussonic Agent 192.168.0.104

Firmware	v20.02.1-1-g6ca5305
SN	267ec33307a1da0c

4. After the activation process is completed, you will see corresponding message on the screen and will have options to view video from the camera, activate another camera, or go to the app home page.



Add a camera

Device activated

Flussonic Agent 192.168.0.104

Firmware	v20.02.1-1-g6ca5305
SN	267ec33307a1da0c
Connected	Online

Note

It may take some time for a newly connected camera to start sending frames. In practice, this time does not exceed one minute.

ADDING AN ONVIF CAMERA

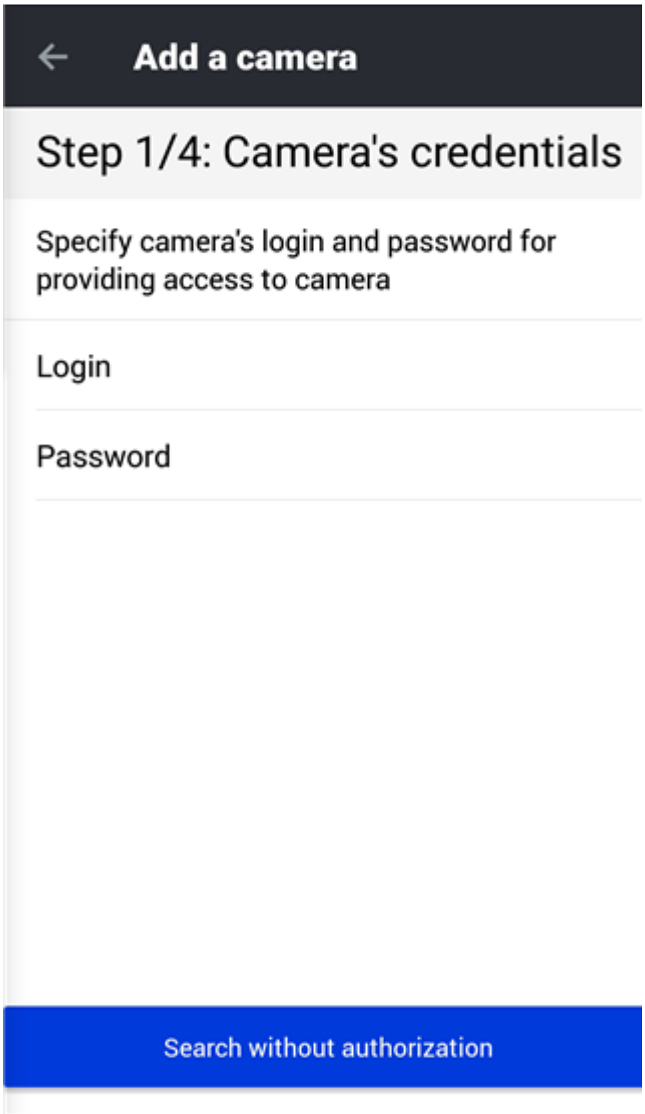
Adding an ONVIF camera in the mobile app utilizes the same principle as camera search in web UI.

Warning

Unlike the above ways of adding cameras via the Watcher mobile app, ONVIF cameras are searched in the local network of **Watcher managing server** (not your phone).

To add an ONVIF camera:

1. Select **ONVIF camera** in the add camera menu.
2. Enter credentials to access the camera via ONVIF or tap **Search without authorization** if the camera can be connected without authorization. Please remember that ONVIF credentials may differ from camera credentials.

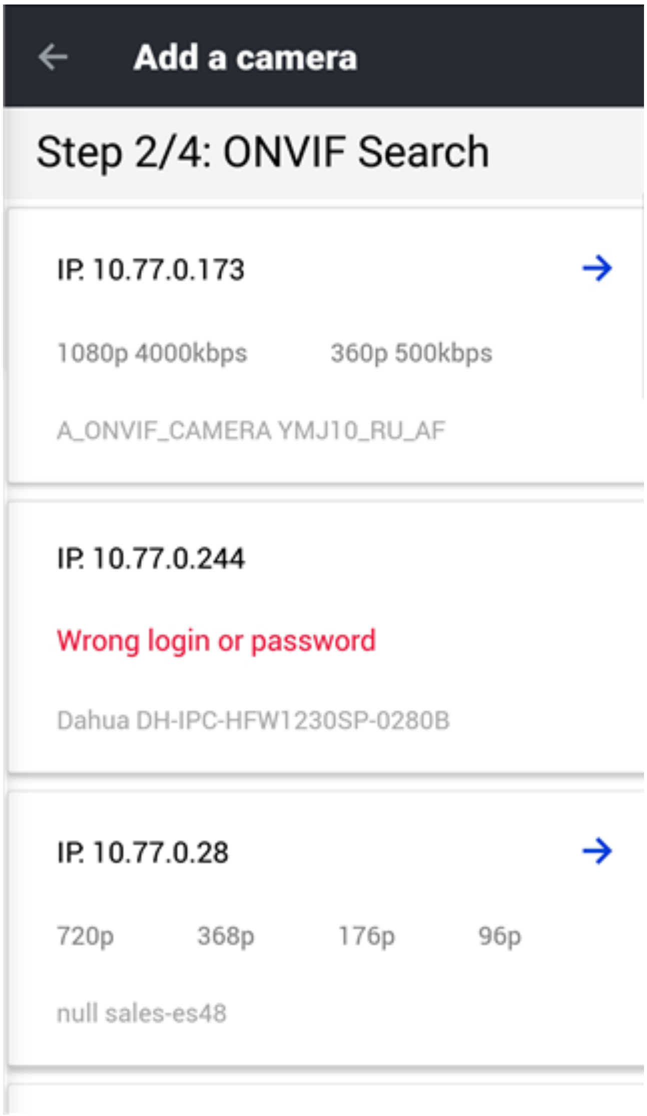


Add a camera

Step 1/4: Camera's credentials

Specify camera's login and password for providing access to camera

3. The found cameras will be displayed in the app. Tap the camera you want to add.



Add a camera

Step 2/4: ONVIF Search

IP. 10.77.0.173

1080p 4000kbps 360p 500kbps

A_ONVIF_CAMERA YMJ10_RU_AF

IP. 10.77.0.244

Wrong login or password

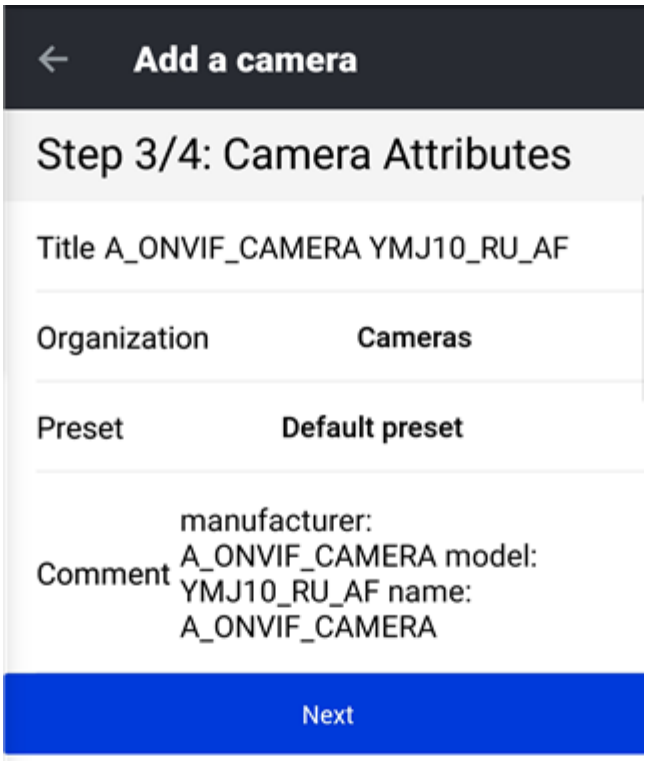
Dahua DH-IPC-HFW1230SP-0280B

IP. 10.77.0.28

720p 368p 176p 96p

null sales-es48

4. Fill in the camera information (see item 3 in the above section for details) and click **Next**.



Step 3/4: Camera Attributes

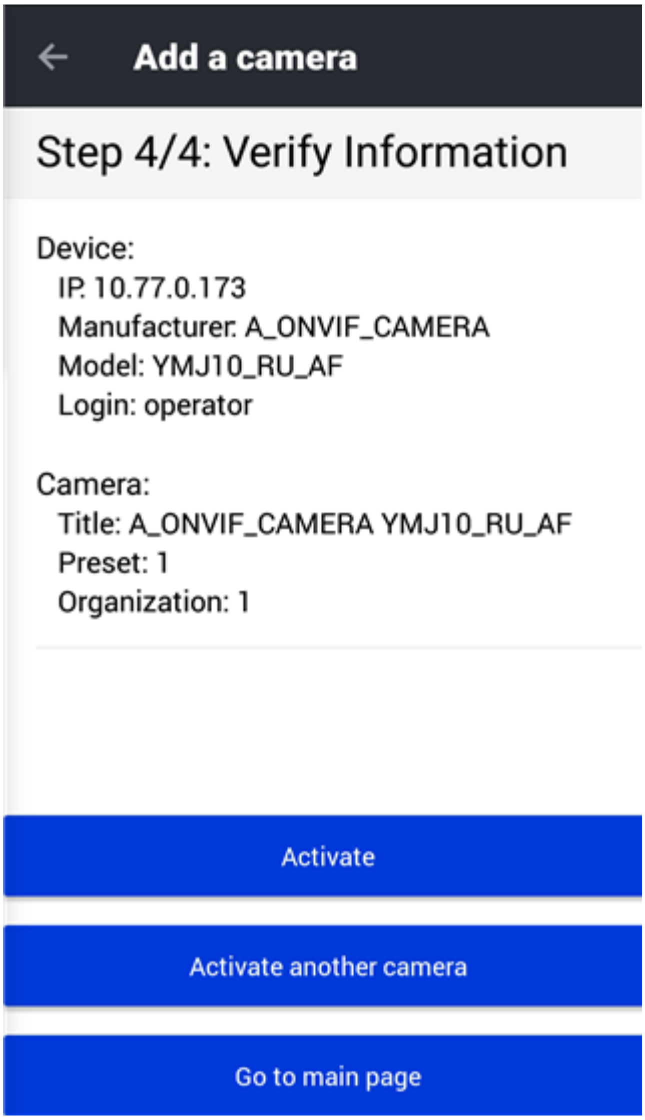
Title A_ONVIF_CAMERA YMJ10_RU_AF

Organization Cameras

Preset Default preset

Comment manufacturer: A_ONVIF_CAMERA model: YMJ10_RU_AF name: A_ONVIF_CAMERA

5. Check the camera info and tap **Activate**.

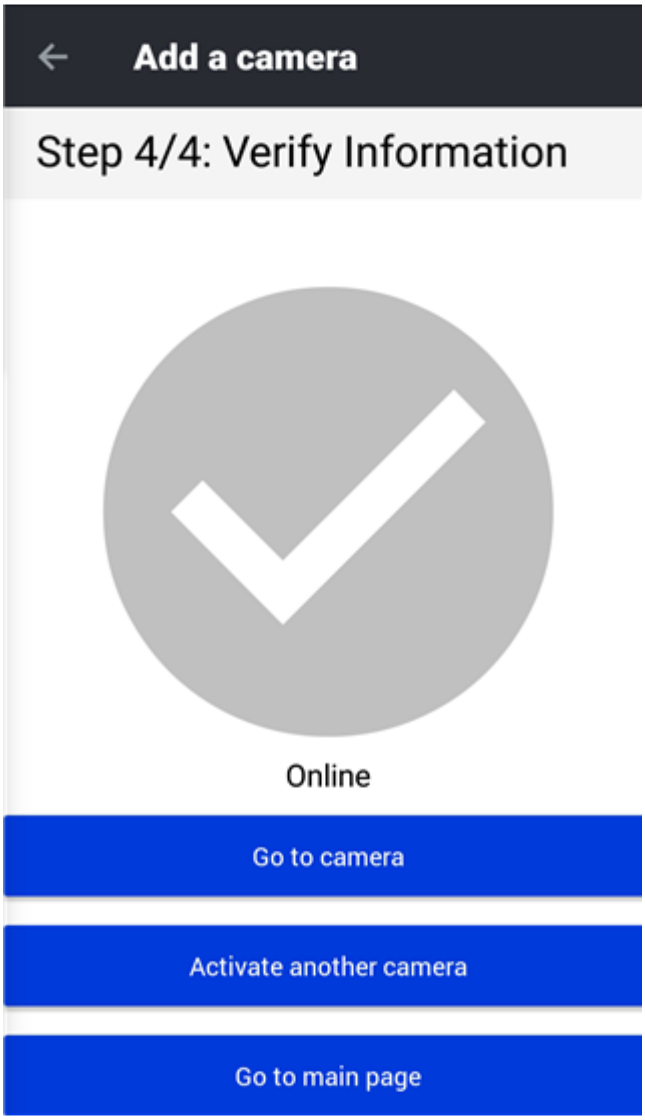


Step 4/4: Verify Information

Device:
IP. 10.77.0.173
Manufacturer. A_ONVIF_CAMERA
Model: YMJ10_RU_AF
Login: operator

Camera:
Title: A_ONVIF_CAMERA YMJ10_RU_AF
Preset: 1
Organization: 1

6. After the activation process is completed, you will see corresponding message on the screen and will have options to view video from the camera, activate another camera, or go to the app home page.



Note

It may take some time for a newly connected camera to start sending frames. In practice, this time does not exceed one minute.

Deleting a camera

To delete a camera:

1. In the app: select the camera and tap **Delete** on the **Actions** tab. You may need to enter the camera name for confirmation.
2. In the web UI: in the 'Cards' mode, click the trash bin icon on the camera preview; in the 'List' and 'Dashboard' modes, select the **Delete** option in the menu on the right of the camera name. See this page for details on the display modes. Another way to delete a camera is by clicking **Delete** on the camera settings page.

4.3.4 Adding Cameras to Folders

If an Organization has a large number of added cameras, it becomes important to have the means to navigate them easily. To address this, Watcher provides *Folders*, which are used to group cameras on some basis, for example, based on their geographical location.

One folder can include a number of subfolders and cameras. Placing cameras into folders is similar to organizing the storage of documents in the file system.

The folders are also required for the following purposes:

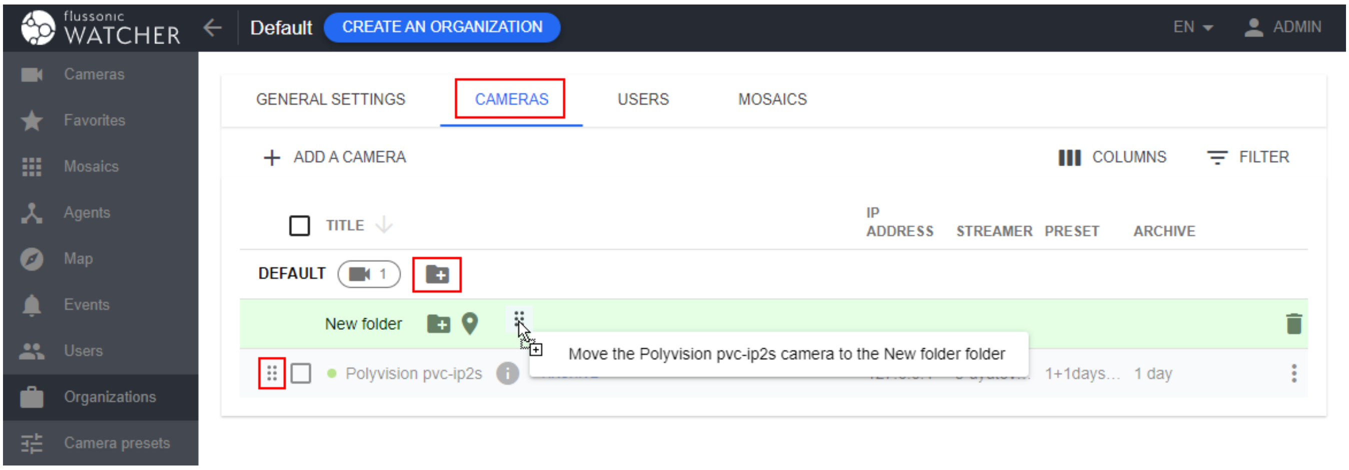
- To create layered graphics plans.
- To manage permissions to cameras. The folder is a minimum permission unit, i.e. you can set each user's permissions to each folder but not to individual camera.

Creating a folder and adding cameras to it

By default, all cameras in an Organization are added to the root folder.

To create a folder and add cameras to it:

1. Go to **Organization** in the Watcher UI
2. Open the Organization profile and select the **Cameras** tab
3. Click the **Add a folder** icon next to the Organization name
4. Drag a camera from the list to the folder using the six-dot control on the left.



Move the Polyvision pvc-ip2s camera to the New folder folder

Note

To rename a folder, just click its name.

Another way to add a camera to a folder is by editing the **Folder** field in the camera settings.

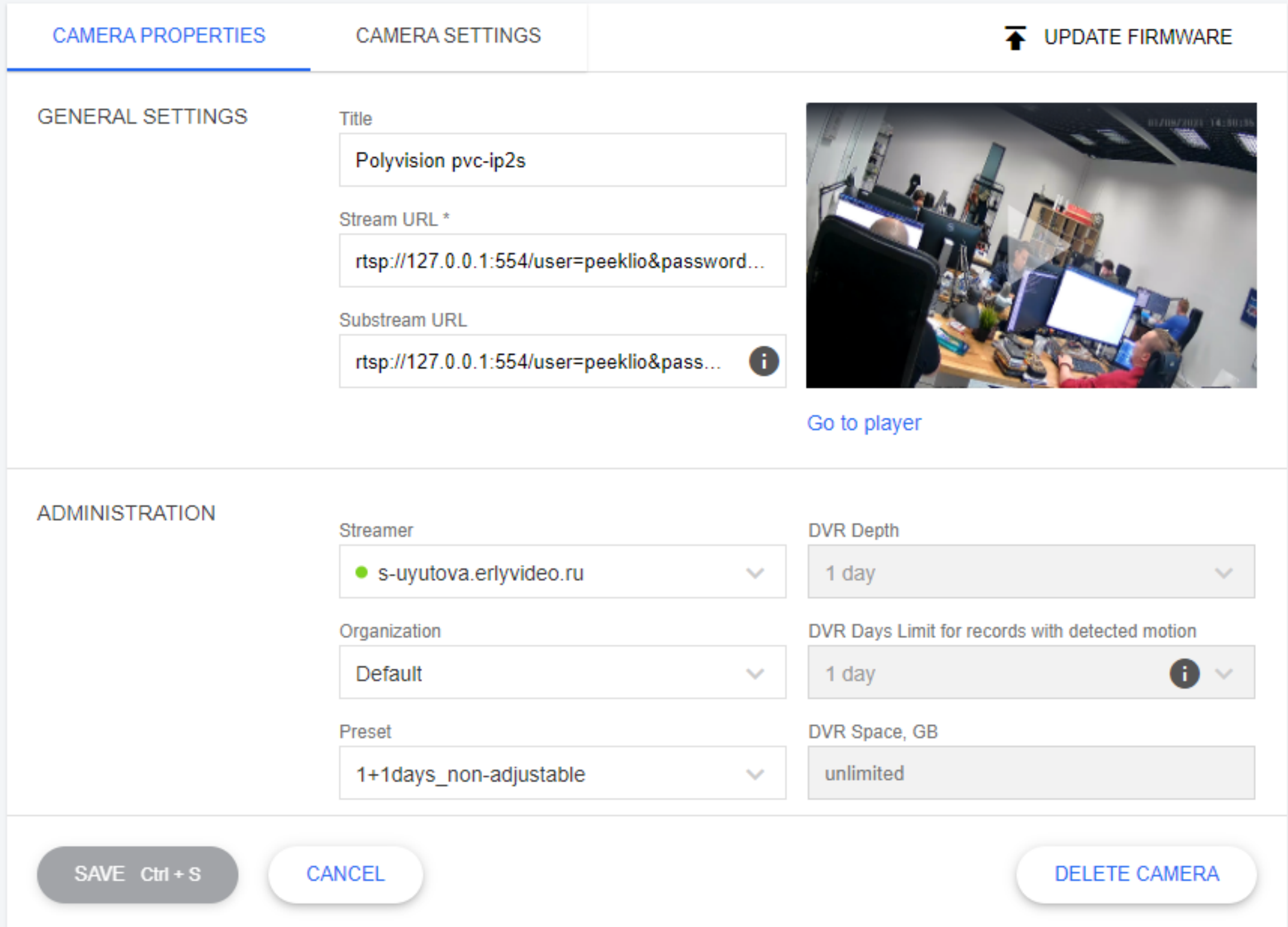
Deleting folders

To delete a folder:

1. Move all the cameras from the folder you are going to delete. You can move them to another folder or to the root folder of the Organization.
2. Click the trash bin icon next to the folder.

4.3.5 Camera settings

Having added and arranged your cameras, you might want to change some of their settings in Watcher. All the available parameters are described below.



Title: Polyvision pvc-ip2s

Stream URL *: rtsp://127.0.0.1:554/user=peeklio&password...

Substream URL: rtsp://127.0.0.1:554/user=peeklio&pass...

Streamer: s-uyutova.erlyvideo.ru

Organization: Default

Preset: 1+1days_non-adjustable

DVR Depth: 1 day

DVR Days Limit for records with detected motion: 1 day

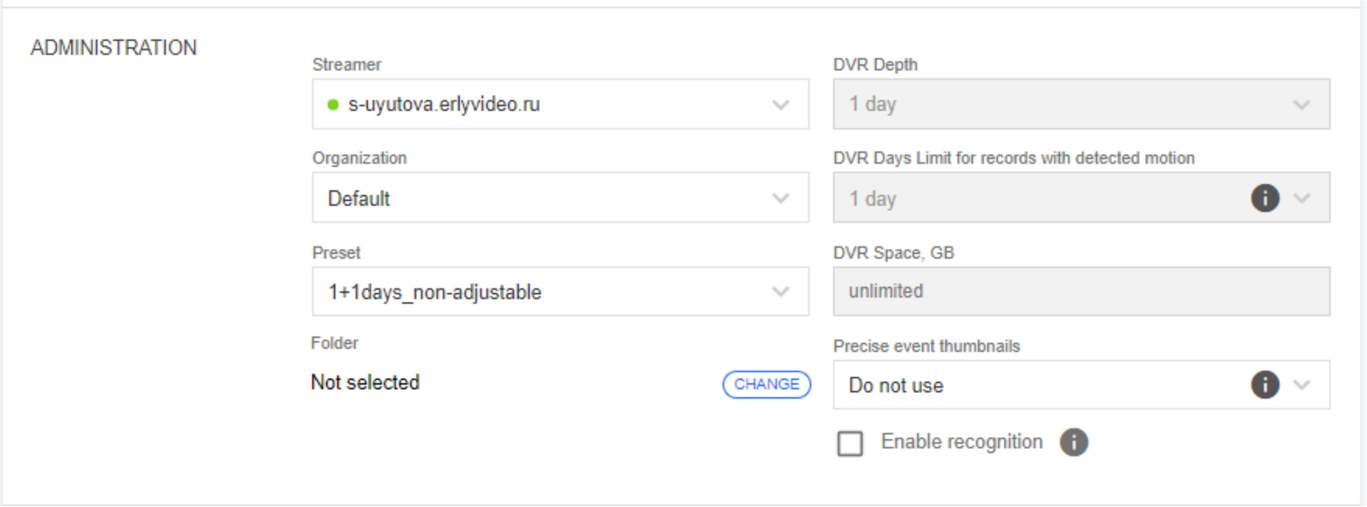
DVR Space, GB: unlimited

See also Camera Remote Setup via ONVIF.

CAMERA PROPERTIES > GENERAL SETTINGS

- **Title**. The camera name, which you see in the list of cameras. Use only Latin characters and numbers, as the name will be used in URLs.
- **Stream URL**. The camera URL. For example: `rtsp://mycam.local/stream0` . If you know the camera URL, you can import the camera from the Flussonic server or use auto-search to add the camera.
- **Substream URL**. Additional address(es). For example: `rtsp://mycam.local/stream1; rtsp://mycam.local/stream2` . The substream parameters are usually different from the main stream, for example, they may have different resolutions or fps. This allows a multi-bitrate stream from the streamer to a client application, i.e. dynamic adaptation of the video stream to the network bandwidth.
- Screenshot of the camera stream (periodically updated). One click starts live video right in the preview box, double click opens the full-screen live player. You can also click **Go to player** to open the DVR player. This is useful if you want to see how new settings applied.

CAMERA PROPERTIES > ADMINISTRATION



ADMINISTRATION

Streamer: ● s-uyutova.erlyvideo.ru

Organization: Default

Preset: 1+1days_non-adjustable

Folder: Not selected

DVR Depth: 1 day

DVR Days Limit for records with detected motion: 1 day

DVR Space, GB: unlimited

Precise event thumbnails: Do not use

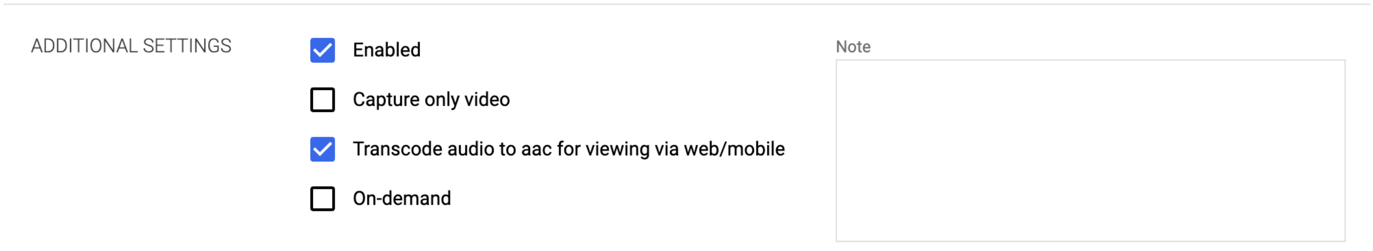
□ Enable recognition

- **Streamer**. A Flussonic Media Server that is used as a streamer for this camera.

> **Note**
>
> If the camera was connected to one streamer for some time and then you change the streamer, the archive already recorded by the camera remains on the old streamer. The new archive is recorded to the new streamer. However, the user (subscriber) will be able to access both new and old DVR seamlessly without having to think about where the archive is actually stored. This feature is called "seamless archive" or "seamless DVR".
>
> The streamers can also switch automatically in the Failover mode.

- **Organization**. The Organization to which the camera will be added. A camera can belong to only one Organization. If you do not select Organization, the default Organization is used.
- **Folder**. A hierarchical node of the camera tree within the Organization. Tha camera will be added to this node called folder. If you do not select a folder, the root folder in the selected Organization is used.
- **Preset**. Select a previously created preset with DVR settings that will be applied to this camera.
- **DVR depth**, **DVR days limit for records with detected motion**, **DVR space**, **Precise event thumbnails** are the archive parameters. They are unavailable and/or overwritten when you select a non-adjustable preset. Learn more at Archive settings.
- **Enable recognition**. Turning on or off license plate detection and recognition (LPR) OR face recognition. This setting is unavailable and/or overwritten when you select a non-adjustable preset. When the box is checked, you can select the recognition type in the drop-down list under it as well as click the button to set the detection zone.

CAMERA PROPERTIES > ADDITIONAL SETTINGS



ADDITIONAL SETTINGS

☑ Enabled

□ Capture only video

☑ Transcode audio to aac for viewing via web/mobile

□ On-demand

Note

- **Enabled**. Turning the camera on and off. It means whether video from this camera will be transmitted or not to Watcher.
- **Capture only video**. When the box is checked, audio stream from camera is *not* captured on streamers nor recorded to DVR or played at live video. When unchecked, i.e. by default, the streamer captures both video and audio streams (if available).
- **Note**. The text that describes camera positioning or gives any other information.
- **Transcode audio to AAC**. Transcode audio to AAC for viewing via web/mobile.
- **On-demand**. Turn this check box on to make the camera transmit video only on request, turn off to make the camera work constantly. This mode of camera operation allows to reduce the load on streamers, but it has a number of significant limitations. Please note that when this box is checked, the video does not come from the camera to the streamer until some action is taken that is considered a video request: this can be viewing the video in Watcher app or web UI, using a direct link to watch video from the camera or just opening the **Cameras** page. If you have turned on the On-demand mode and configured motion events, analytics (face recognition or LPR) and/or DVR, these functions will only work when the video is received by the streamer. In other words, if no one requests a stream from the camera, then no functions associated with it work. Hence, it is recommended to check this box with caution, only in cases that do not require any functions other than viewing live video from the camera (e.g. for intercom, security desk camera, etc.).

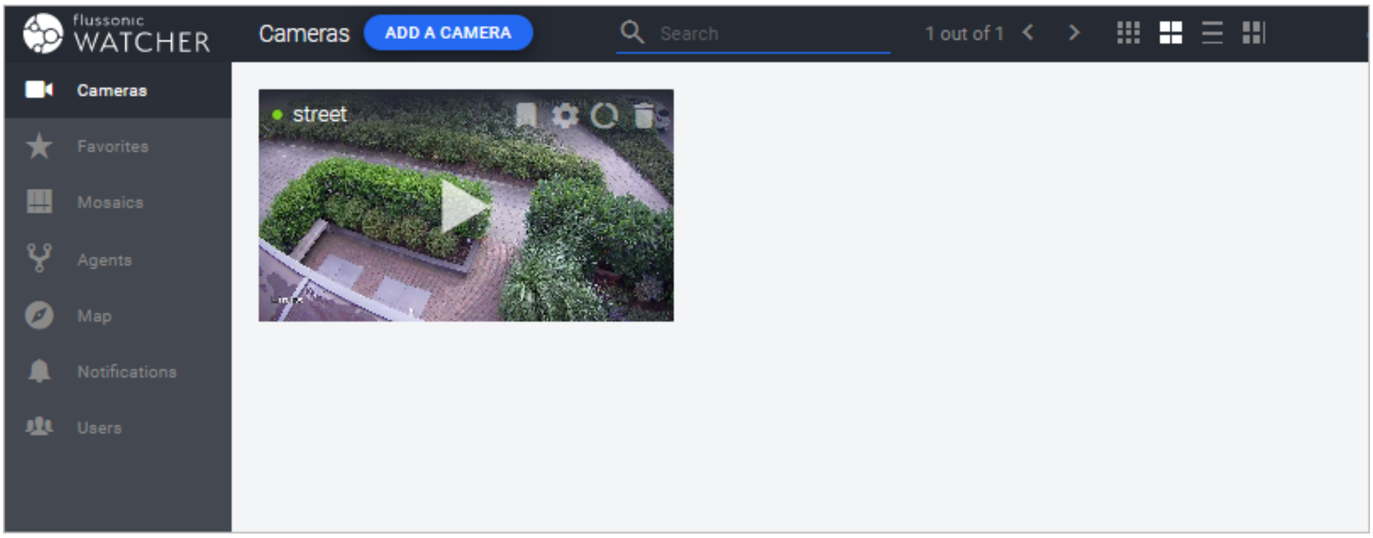
  The timeout for disconnecting the camera after disconnecting the last client is 60 seconds.

CAMERA PROPERTIES > LOCATION

- **Coordinates**. The coordinates of the camera location. Define the camera placement on the map. You can change its placement. We recommend that you place all cameras to their actual locations on the map to help users find them.
- **Address**. The postal address of the camera.

SAVING CAMERA SETTINGS

After you have edited the camera settings, click **Save**. The camera will appear on the management page in the list of cameras:



4.3.6 Archive settings

Video archive storage is a fundamental function of any video surveillance system. Archive recording allows for incidents analysis and violations discovery, as well as helps in law enforcement intelligence.

On this page, you will find out how to configure the archive recording. The archive in Watcher can be recorded continuously or on motion/analytics events. Motion-based recording saves disk space if there is little activity in the camera's field of view. You can also limit the disk space allocated for the archive so as not to fill the disk completely which can disrupt the operating system.

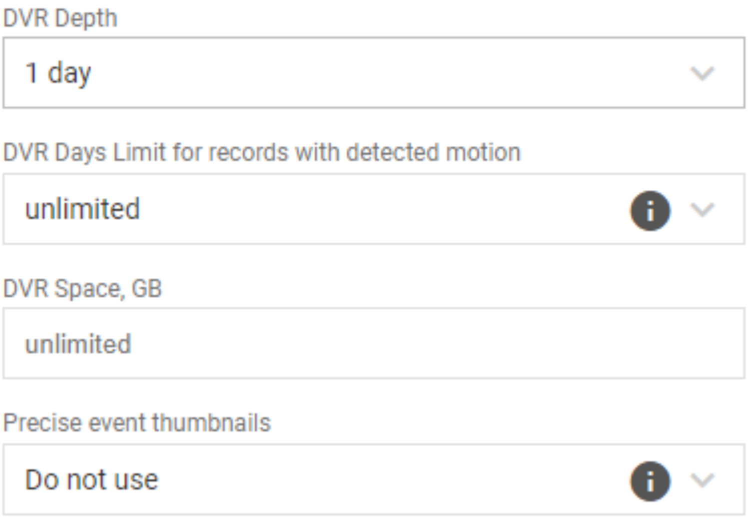
See also:

- Viewing the camera's live video or DVR archive about viewing the archive and exporting records.
- Events about event list and searching for events.

Archive parameters

Setting up an archive in Watcher is convenient with presets. In a preset, you can set a template for archive parameters, and then select it on cameras as needed. If the preset is customizable, then parameters can be changed later individually for each camera.

You can set the following archive parameters in the preset and/or in the camera settings:



DVR Depth: 1 day

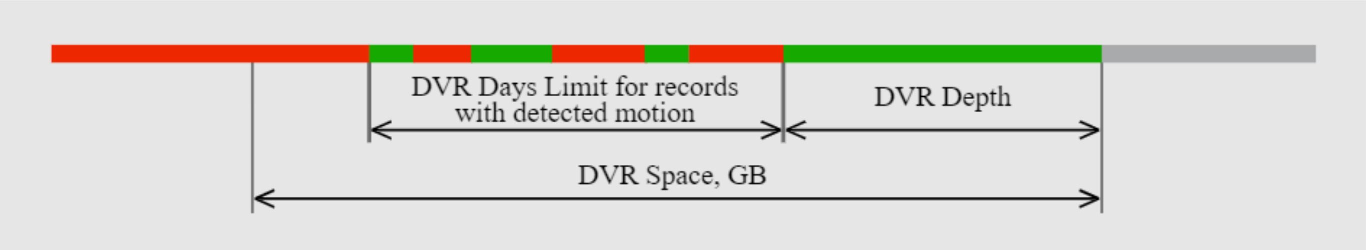
DVR Days Limit for records with detected motion: unlimited

DVR Space, GB: unlimited

Precise event thumbnails: Do not use

- **DVR depth**. The archive depth. In other words, it’s the number of days after recording during which the video archive is available (then it is purged but motion detection records may be left in place according to the setting described in the next item). Available values:
  - **DVR disabled**: archive is not recorded.
  - **Record on motion**: only motion events are recorded.
  - **The number of days/weeks/months**: the archive is recorded continuously and stored for the time specified in the parameter. If a longer period is selected for motion recording, then motion recordings are stored longer.
- **DVR days limit for records with detected motion**. The number of days during which it is necessary to keep records of motion events. This limit is set in addition to the archive depth. If you select **not set**, then all motion events are stored. Available values:
  - **Unlimited**: motion event recordings are stored until the maximum archive size (**DVR space**) is reached or the disk is full.
  - **DVR disabled**: motion event recordings are stored for the same time as the rest of the archive, i.e. as set in **DVR depth**.
  - **The number of days/weeks/months**: motion recordings are stored for the specified time.
- **DVR space**. The maximum storage space for camera's archive, in Gigabytes.

The relationship between DVR parameters is explained in the diagram below:



- **Precise event thumbnails** is a number of days for which the precise screenshots of face recognition and/or LPR events are available. These screenshots are displayed on the **Events** page. The object that triggered the detector is guaranteed to be present on the precise screenshot. If disabled or the precise screenshot storage time expires but there is an archive for the period, then the first frame of the five-second video segment in which the event occurred is displayed on the **Events** page. In this case, the object that triggered the detector may be present at any place in the video fragment, so you will have to search for it manually.

> **Warning**
>
> ```
> The precise screenshots are stored in Flussonic Watcher database. Make sure your system complies with the [requirements](hardware-requirements.md#watcher-system-requirements) before selecting a long storage period.
> ```

Archive cleanup

The archive cleanup process starts automatically once an hour. Records that have expired are deleted as well as the oldest records, if the maximum archive size is reached.

Please note that a camera's' archive is not deleted immediately when you delete the camera; it is cleaned in accordance with the archive storage parameters that were set for the camera. For example, if you set an unlimited storage time for motion events recordings, they are deleted only when the DVR space limit is reached or when the disk is full.

4.3.7 Camera Remote Setup via ONVIF

Managing camera settings from the Watcher UI via ONVIF

You can change a camera's settings directly from the Watcher UI. The ONVIF protocol is used for this, so make sure your camera supports it.

On the **ONVIF configuration** tab, Watcher allows you to setup:

- PTZ control and receiving motion events
- The output stream from the camera (resolution, frame rate, codec, etc.)
- The network settings of the camera (DHCP, static IP address)
- Image settings (brightness, sharpness, etc.)
- Time setting (add an NTP server, set time manually)
- User accounts.

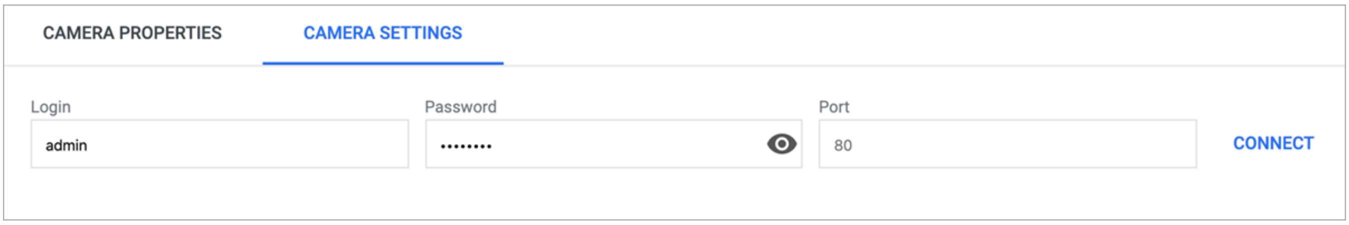
The details are described below on this page.

> **Note**
>
> If some settings are disabled (not editable), then the camera model does not support their change (except the IP address, it can be disabled when you use DHCP).

TO CONFIGURE A CAMERA REMOTELY:

1. Go to the camera settings: you can do so from the **Cameras** page or from the **Organizations** page
2. Open the **ONVIF configuration** tab
3. If the camera is disconnected, enter the credentials configured on the camera and click **Connect**

> **Warning**
>
> Both login and password must NOT include any of the following characters: @, ;, #, [, \, /, =, $



Login: admin

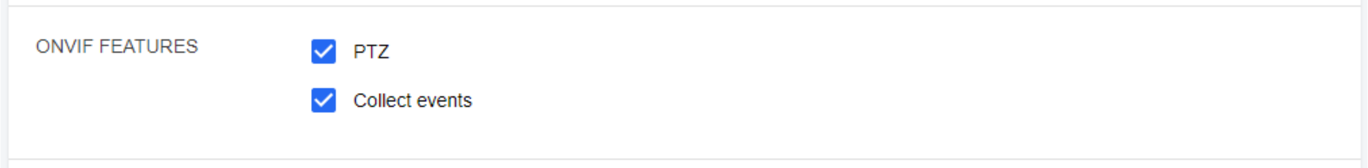
Password: ••••••••

Port: 80

4. After the camera has been connected, the UI for managing it via ONVIF appears.

ONVIF FEATURES

If the camera connected via ONVIF supports PTZ control and/or receiving motion events, you can enable or disable these functions from the Watcher web UI.



ONVIF FEATURES

☑ PTZ

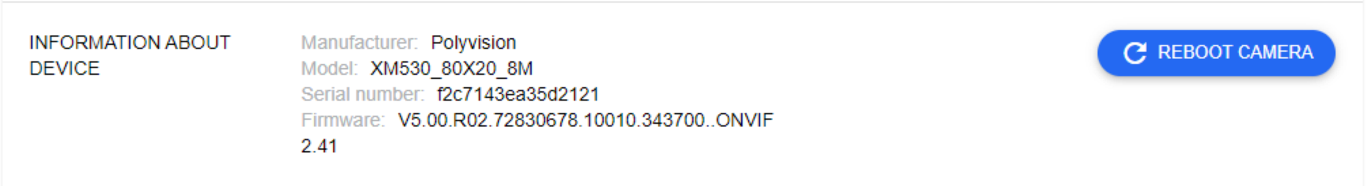
☑ Collect events

- **PTZ**. Turn this check box on for cameras with PTZ feature to enable camera PTZ controls in the UI.
- **Collect events**. Enables collecting motion events from a camera via ONVIF. You should enable motion detection in the camera's web interface. You can view detected events on the **Events** page.

> **Note**
>
> Watcher can also receive events via SMTP. However, this function requires adjustment on the camera itself, which can be difficult with a large number of cameras. We recommended receiving events via ONVIF. Please do not hesitate to contact our technical support team in case of any troubles with events collection via ONVIF.

INFORMATION ABOUT THE DEVICE

You can see camera details in the Watcher web UI: manufacturer, model, serial number, firmware. These details are returned by the camera and not editable.



INFORMATION ABOUT DEVICE

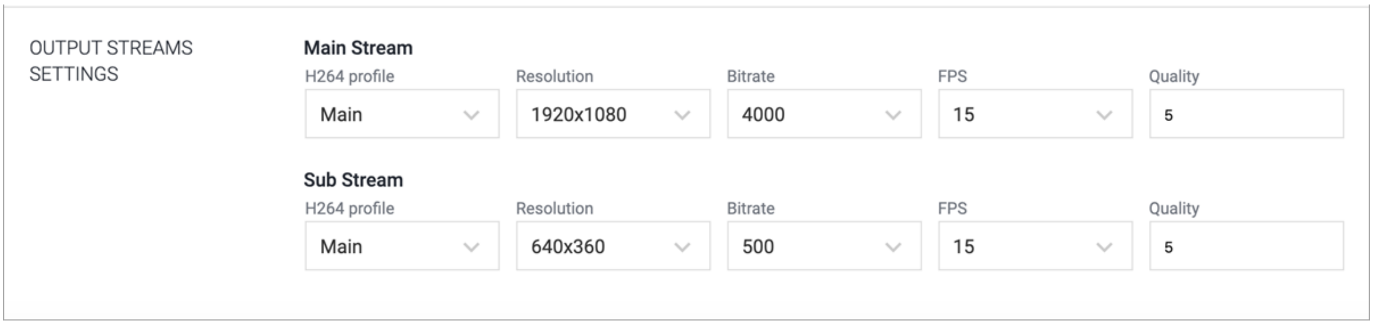
Manufacturer: Polyvision
Model: XM530_80X20_8M
Serial number: f2c7143ea35d2121
Firmware: V5.00.R02.72830678.10010.343700..ONVIF 2.41

To send a command to reboot the camera, click **Reboot camera** in the **INFORMATION ABOUT DEVICE** section.

OUTPUT STREAM SETTINGS

Here you can configure streams provided by the camera. The camera may have one or more streams depending on the model. Some stream settings can depend on each other, e.g. you may have to set resolution or fps according to certain rules. This also depends on the model. Please refer to the camera documentation for details.

You can specify one or two of the configured streams in the camera properties in Watcher.



OUTPUT STREAMS SETTINGS

Main Stream

H264 profile	Resolution	Bitrate	FPS	Quality
Main	1920x1080	4000	15	5

Sub Stream

H264 profile	Resolution	Bitrate	FPS	Quality
Main	640x360	500	15	5

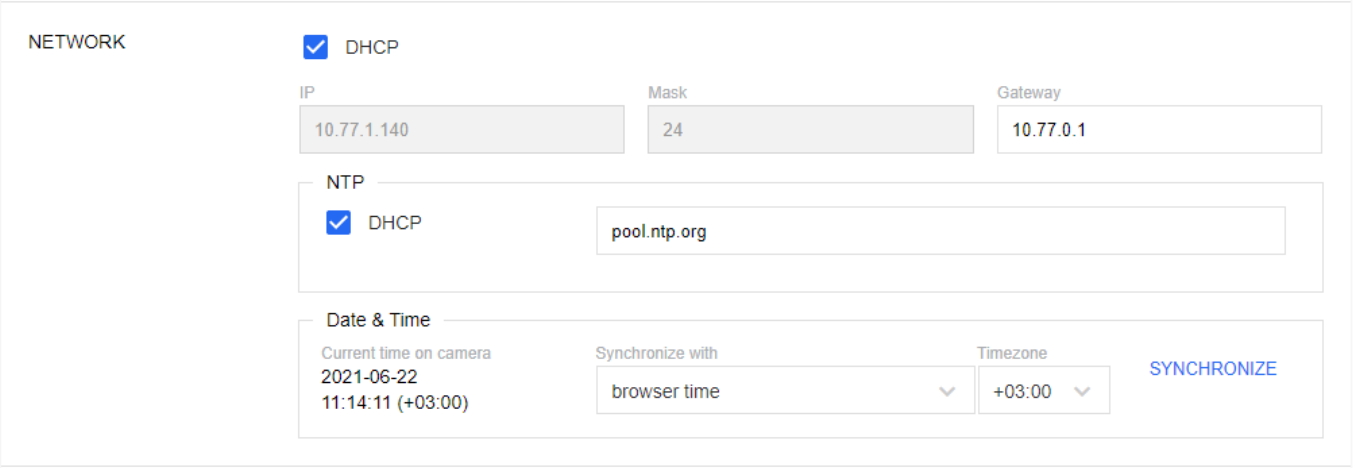
The following settings are available:

- **H264 Profile**: select one of the standard H264 profiles
- **Resolution**: set the video resolution
- **Bitrate**: stream bit rate, kbps
- **FPS**: frames per second
- **Quality** is the quality of the image from the camera, in units from 0 (low quality) to 5 (high quality). The higher the quality, the higher the bitrate of the stream.

NETWORK

You can configure camera's network settings or synchronize the camera time with NTP server or you browser time.

Time synchronization is necessary for correct time stamping. Timestamps are used for archive playback to display archive for the time period you select.



NETWORK

☑ DHCP

IP: 10.77.1.140

Mask: 24

Gateway: 10.77.0.1

NTP

☑ DHCP

pool.ntp.org

Date & Time

Current time on camera: 2021-06-22 11:14:11 (+03:00)

Synchronize with: browser time

Timezone: +03:00

The following settings are available:

- **DHCP**: check the box for the camera to obtain the IP address and subnet mask settings automatically from the router to which it is connected. If DHCP is used, you cannot manually set the IP address and mask.
- **IP** and **Mask**: you can set the camera IP address and subnet mask if necessary.

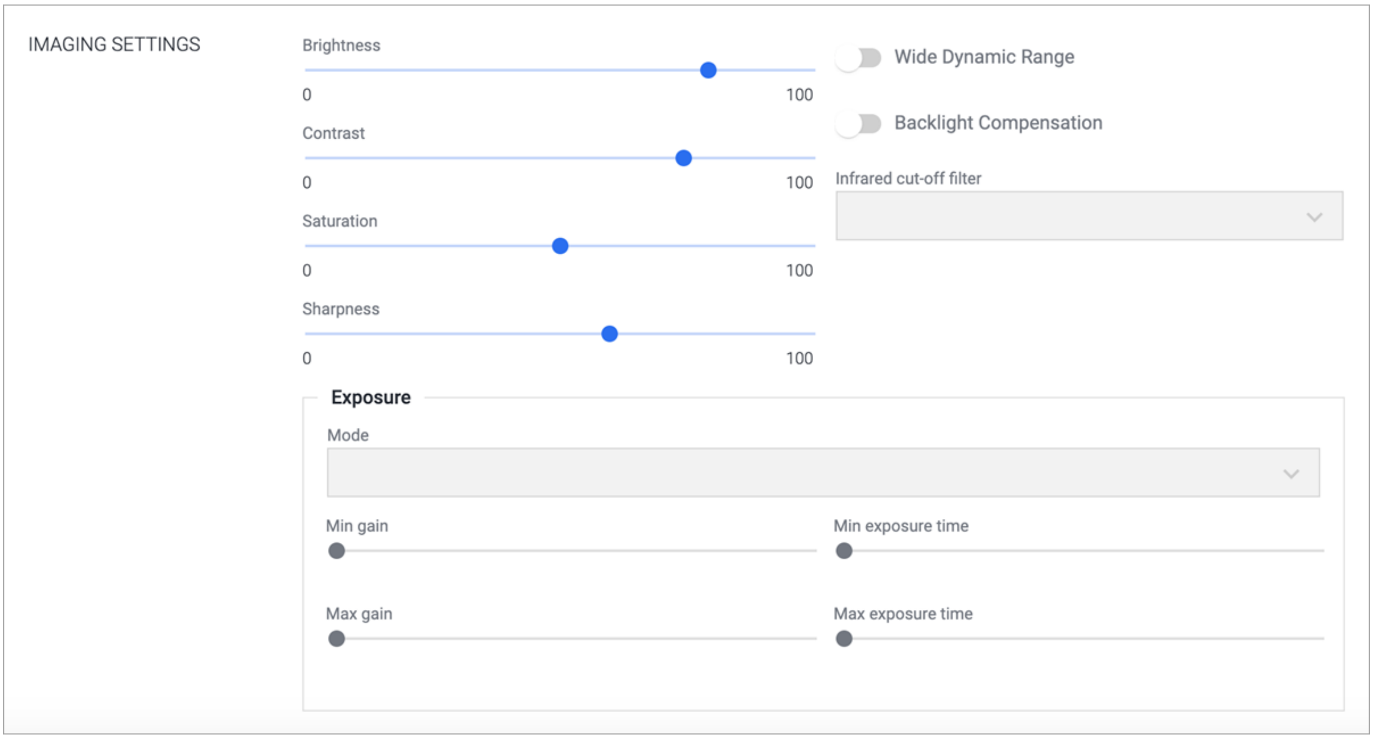
> **Danger**
>
> Be careful when changing these camera settings to avoid losing access to it. If the camera is no longer available on the Watcher server, you will not be able to continue remote configuration.

> **Warning**
>
> To apply any of the network settings, click **Reboot camera** in the **INFORMATION ABOUT DEVICE** section. All other settings take effect without rebooting the camera.

- **Gateway**: set the gateway address.
- **NTP** settings group: check the **DHCP** box in this group for the camera to obtain the address of the time synchronization server (NTP) automatically from the router, OR uncheck the bof and enter the NTP server address in the text field.
- **Date & Time** settings group: check the current camera date and time; if necessary, forcibly set the browser time or NTP server time on camera.

IMAGING SETTINGS

You can change video image settings.

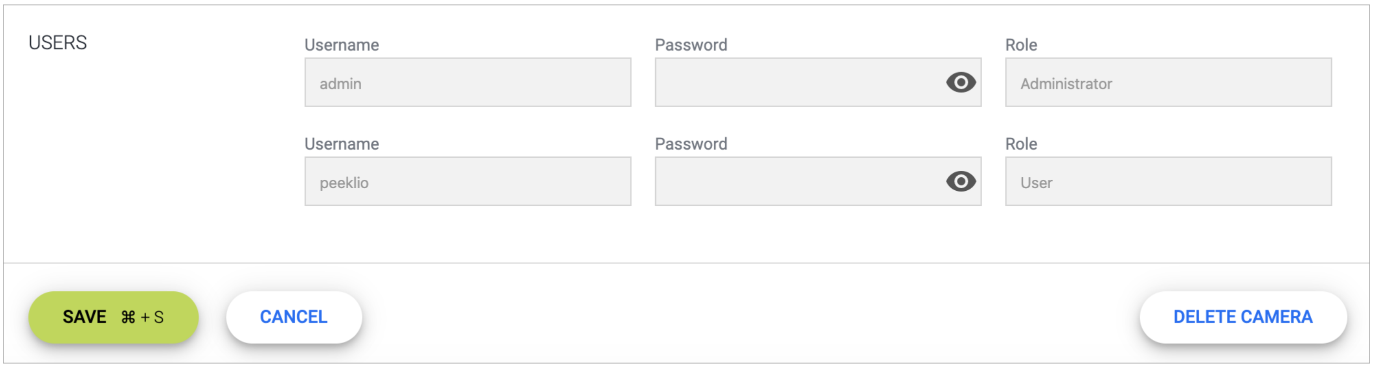


The following settings are available:

- **Brightness**
- **Contrast**
- **Saturation**
- **Sharpness**
- **Wide Dynamic Range**
- **Backlight Compensation**
- **Infrared cut-off filter**
- The **Exposure** settings:
    - **Mode** of the exposure control
    - **Min gain**
    - **Max gain**
    - **Min exposure time**
    - **Max exposure time**

USERS

You can add, delete or edit camera user accounts. These accounts can be used for camera connection, for example, as shown above.



USERS

Username	Password	Role
admin		Administrator
peeklio		User

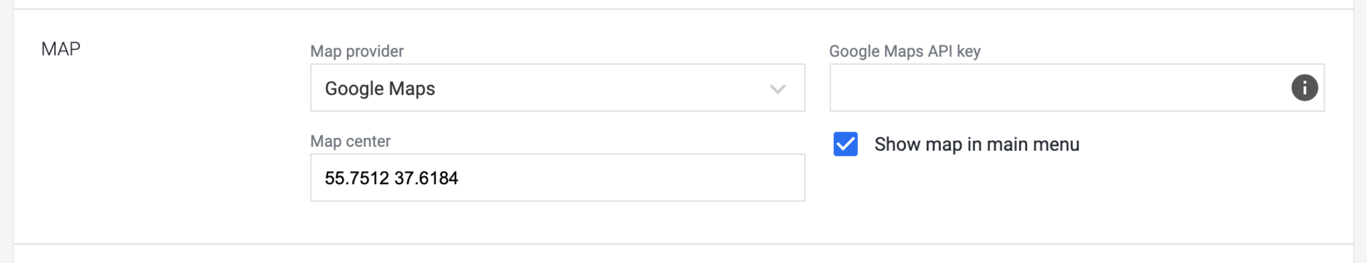
4.3.8 Layered Graphics Plans

Flussonic Watcher can display surveillance cameras on geo maps, as well as combine a geo map and floor plans (layouts) of the facility (premises). This feature allows you to conveniently navigate a large number of geographically distributed cameras, placing them directly on the map and placing them on the layouts of the facility. The solution of providers such as Google Maps, Yandex Maps, OpenStreet Maps can be used as graphic maps, and pictures in JPG and PNG format can be used as floor layouts.

GRAPHIC MAP SETTINGS

Before you start using graphics maps, you must configure them. To do this, go to the **Settings** section, where you will see the **Map** section, on which you are offered to choose:

- Map provider. A service that provides a graphic background for a geo map. Google Maps, Yandex Maps, OpenStreet Maps are supported. For OpenStreet Maps it is also possible to specify an alternative server.
- Center of the map. The point that will open by default when you open the **Map** tab.
- API Key Google Maps. If you use maps from Google, then it is possible to specify the API Key in order to automatically recognize the addresses that are entered in the coordinate field and convert it to coordinates.
- The option to display maps in the main menu. If this option is enabled, the tab with maps will appear in the left navigation menu.



MAP

Map provider: Google Maps

Google Maps API key:

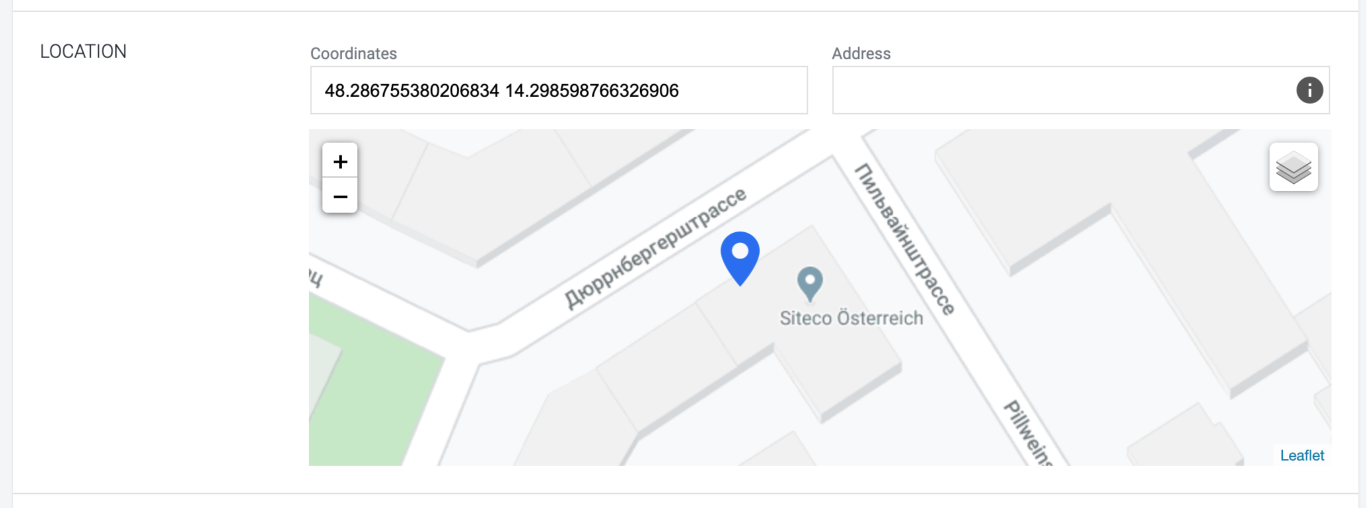
Map center: 55.7512 37.6184

☑ Show map in main menu

ADDING A CAMERA ON A GRAPHIC MAP

In this mode, the camera image will be placed directly on the facility graphic map. This mode is relevant for cases when a small number of surveillance cameras are installed at one facility or when a camera is part of a city video surveillance system.

To place the camera on a graphics map, go to the camera settings, scroll from to the **Location** section and click the point on the map where the camera is physically located. The coordinates of the camera will appear in the appropriate boxes automatically.



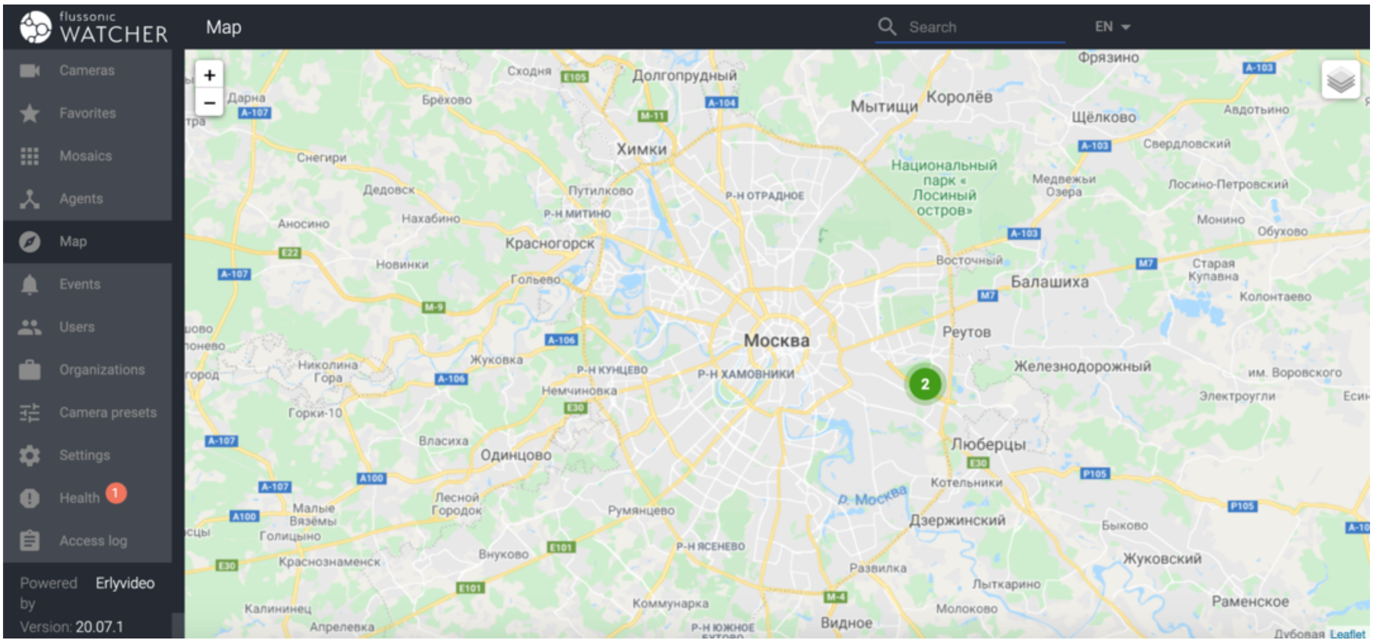
LOCATION

Coordinates: 48.286755380206834 14.298598766326906

Address:

In addition, you can fill the camera coordinates by copying them from the map. After that, when opening the graphics map, you will see an icon corresponding to the installed camera.

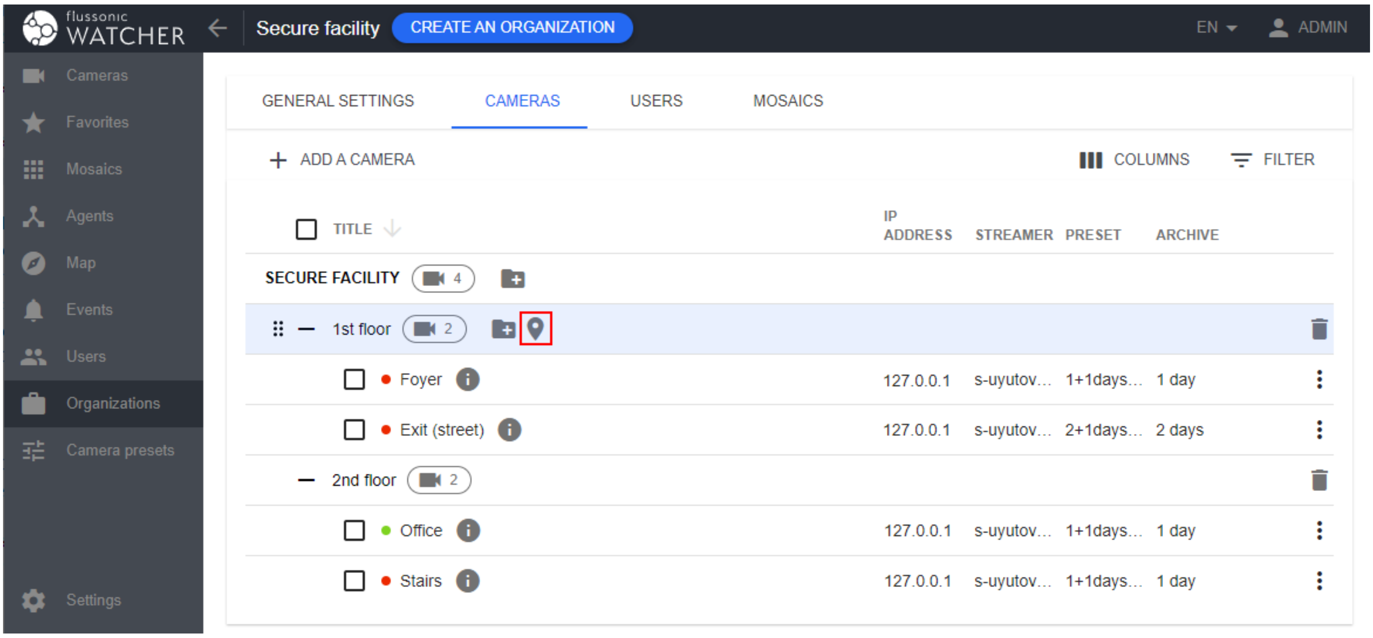
Several cameras can have the same coordinates. In this case, an icon with a number corresponding to the number of cameras placed at this point will appear on the map. To view the picture from a particular camera, click the icon with the number.



ADDING FACILITY LAYOUTS ON A GRAPHIC MAP

You can add the layout of the facility directly to the graphics map and fit it to the size of the image of the facility itself. Let's say a house is displayed on a graphic map, and you have a floor plan of this house. You can setup the map so that at zooming in you could see the layout of this house and its floors.

An important point is that facility layouts are attached to camera folders in the Watcher structure. That is, if you want to place the layout of the house by floors, then a folder in the camera structure must be assigned to each floor and cameras must be added to those folders. The folders structure will be like that:

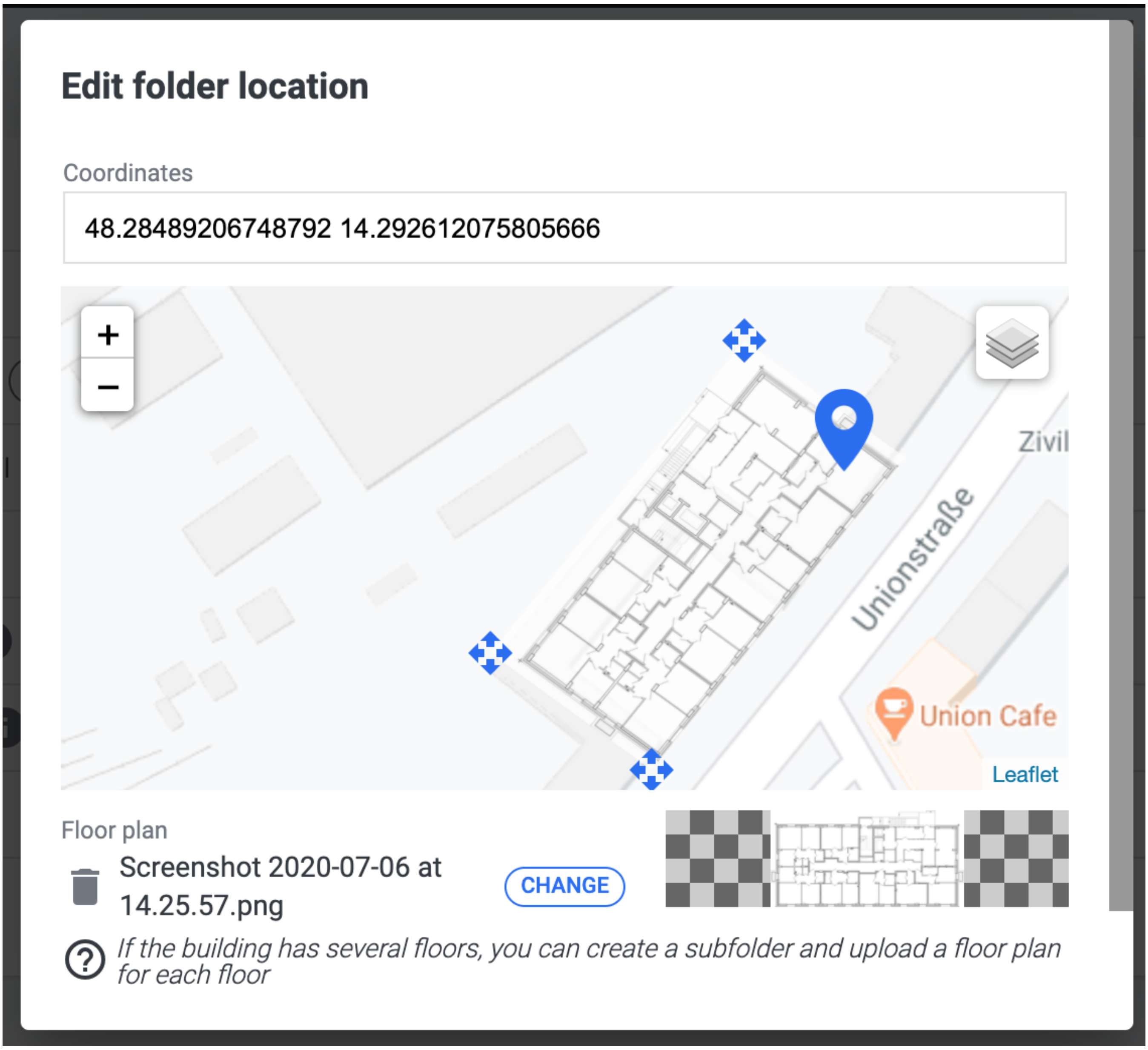


TITLE	IP ADDRESS	STREAMER	PRESET	ARCHIVE
SECURE FACILITY 4				
1st floor 2				
□ ● Foyer	127.0.0.1	s-uyutov...	1+1days...	1 day
□ ● Exit (street)	127.0.0.1	s-uyutov...	2+1days...	2 days
2nd floor 2				
□ ● Office	127.0.0.1	s-uyutov...	1+1days...	1 day
□ ● Stairs	127.0.0.1	s-uyutov...	1+1days...	1 day

To attach the facility layout to the graphic map, click the coordinate icon to the right of the folder name.

In the form that opens, specify the following:

- Specify the center of the facility layout in relation to the graphic map. Just click on the house on the graphic map to which you want to attach the floor plan to specify the coordinates of the center.
- Upload a picture with the layout of the facility
- Stretch the layout of the room along the contour of the building on the graphic map using the appropriate sliders.



Edit folder location

Coordinates: 48.28489206748792 14.292612075805666

Floor plan: Screenshot 2020-07-06 at 14.25.57.png

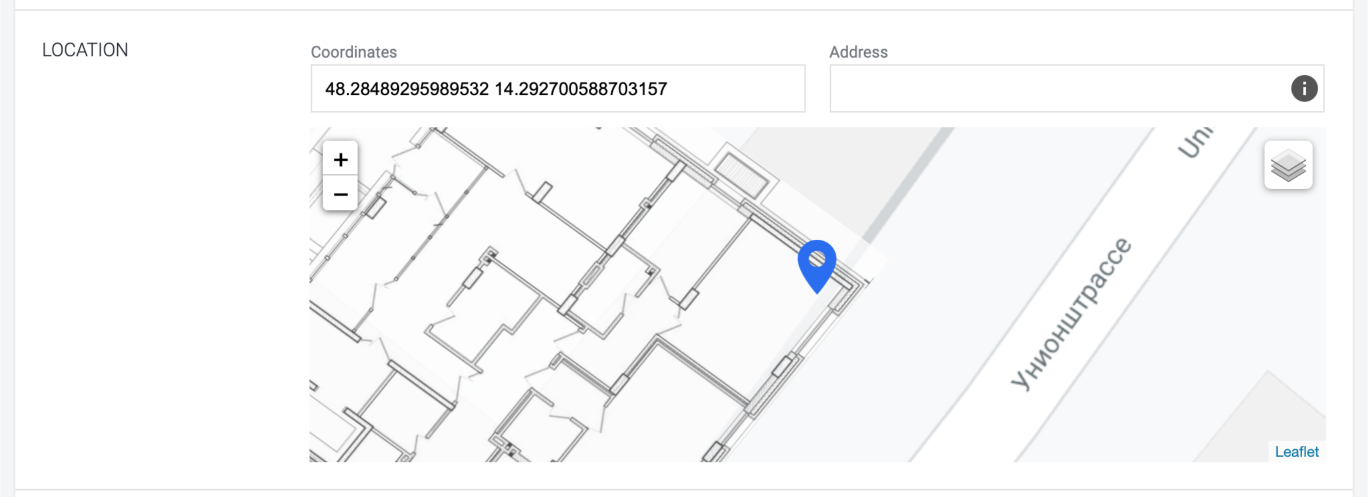
If the building has several floors, you can create a subfolder and upload a floor plan for each floor

If you need to add the layouts of more floors, then create the appropriate subfolders for them, specify a single center on the graphic map and add the appropriate layouts.

After adding subfolders on the graphics map, you will be able to select the layout of the desired floor from the drop-down list in the top right corner of the map.

ADDING CAMERAS ON THE FLOOR LAYOUTS OF THE FACILITY

Floor plans are loaded. Now you can place cameras on them. To do this, simply open the camera settings and scroll to the Map section. In this section, you will immediately see the layout tied to the folder where the camera lies. You just need to click in the place where the camera is physically located on the floor plan.



LOCATION

Coordinates: 48.28489295989532 14.292700588703157

Address:

USING GRAPHICS MAPS AND LAYERED FACILITY LAYOUTS

Everything is ready and configured. Now you can see how it works. Open the graphic map where you will see cameras and facilities with floor plans attached. Click the icon of such a facility to see its layout with cameras placed on it. If the facility has multi-level plans, then you will see a drop-down list from which you can choose the floor you are interested in.

4.3.9 User and permission management

Standard Watcher users exist within Organizations. One user can be added to several different Organizations and must be added at least to one. If you do not select organization when creating a user, the user is added to the Default Organization.

After installing Watcher, you will see only one user on the **Users** page - the Watcher Super Administrator having all possible permissions. Do not delete this user as it may lead to incorrect operation of the system.

The newly created users can be assigned with flexible permissions within Organizations and Watcher. Please refer to the table below for explanation of available permissions.

On this page:

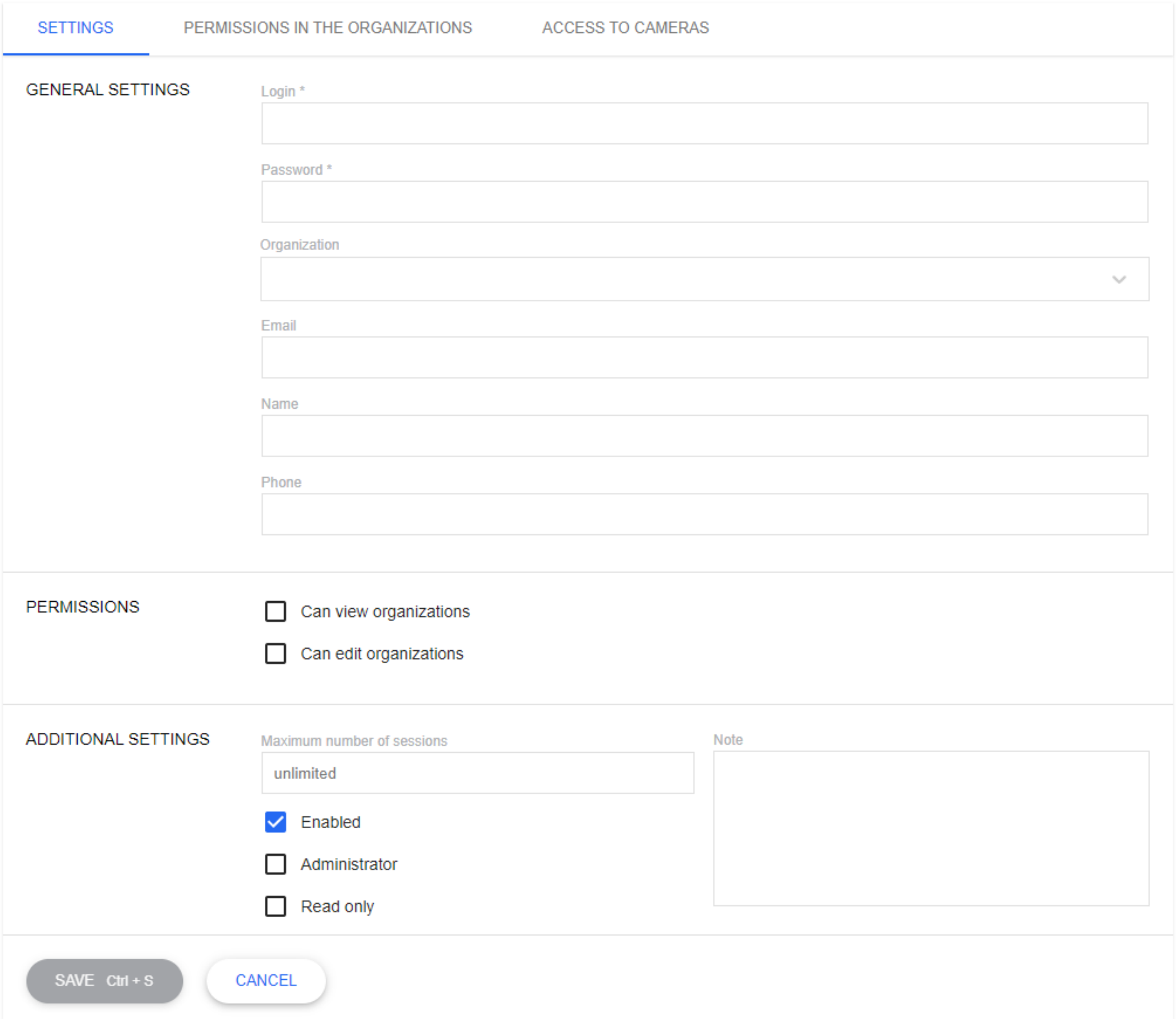
- Creating a user in Watcher
- Using the list of Watcher users
- Adding users to an Organization
- Deleting users from an Organization
- Granting a user rights to manage Organizations
- Granting a user access to cameras

Creating a user

Make sure you already have created the Organization to add user in.

To create a user:

1. Click **Users** — **Create user**.
2. The form to create the user opens.



GENERAL SETTINGS

Login *

Password *

Organization

Email

Name

Phone

PERMISSIONS

☐ Can view organizations

☐ Can edit organizations

ADDITIONAL SETTINGS

Maximum number of sessions: unlimited

☑ Enabled

☐ Administrator

☐ Read only

Note

3. Fill in the user info, change the following settings if necessary:

   - **Can view organizations**: Check the box to enable the user to view the list of all organizations. If the box is not checked, the user will only see the organizations to which the user is added.
   - **Can edit organizations**: Check this box to enable the user to edit the organizations. All Organizations become visible to the user when you check this box, i.e. the **Can view organizations** box is automatically set.

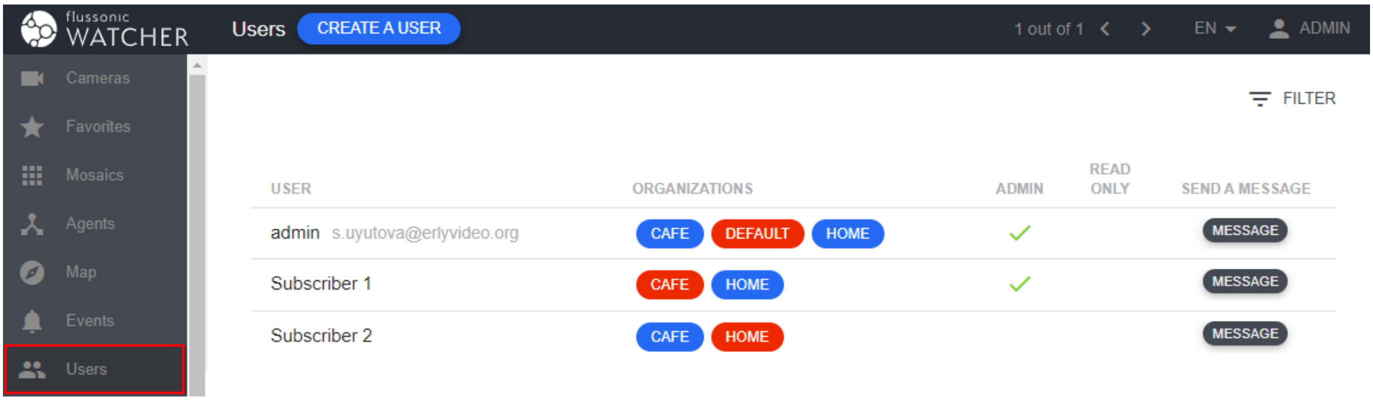
     Editing options differ depending on whether the user is an organization administrator. The user can change all the settings in owned organization(s) and only **General settings** tab in the rest of the organizations.
   - **Maximum number of sessions**: The maximum number of sessions for the user. This setting allows you to limit the use of the same account by different people, for example, if the user shares credentials with someone else. Each time the user logs into the system (i.e. enters their username and password), a unique session ID is saved in the Watcher database. If the specified number of sessions is exceeded, the oldest session ID is deleted and a new one is created.
   - **Enabled**: Check the box to activate the user account.
   - **Administrator**: Check the box to give the user administrator permissions (see details above). When you save the settings after selecting the **Administrator** checkbox, the checkboxes enabling organizations visibility and editing are automatically checked.

     > **Note**
     >
     > Please note that if you uncheck the **Administrator** checkbox, the checkboxes enabling organizations visibility and editing are not unchecked automatically. Uncheck them manually if necessary under the Super Administrator account.

   - **Read only**: Check the box to prohibit the user from any actions with cameras, organizations and users, except for viewing. This checkbox cannot be selected together with the **Administrator** checkbox. If both are checked, the **Administrator** checkbox takes precedence, so the **Read only** checkbox is automatically cleared when you save the user. Otherwise, the **Read only** checkbox takes precedence over the checkboxes that allow editing organizations, users, and cameras.

Watcher user list

The list of all Watcher users is available on the **Users** page.



USER	ORGANIZATIONS	ADMIN	READ ONLY	SEND A MESSAGE
admin s.uyutova@erlyvideo.org	CAFE DEFAULT HOME	✓		MESSAGE
Subscriber 1	CAFE HOME	✓		MESSAGE
Subscriber 2	CAFE HOME			MESSAGE

This list allows you to:

1. Open the user profile for editing or deleting by clicking the user name.
2. View Organizations to which the user is added. The names of the Organization(s) owned by the user are highlighted in red. Click the name of the Organization to view its profile.
3. Check if the user is Watcher Administrator or has Read-Only permissions.
4. Send messages to the user.
5. Search users by their login. Click **FILTER** to open the search form.

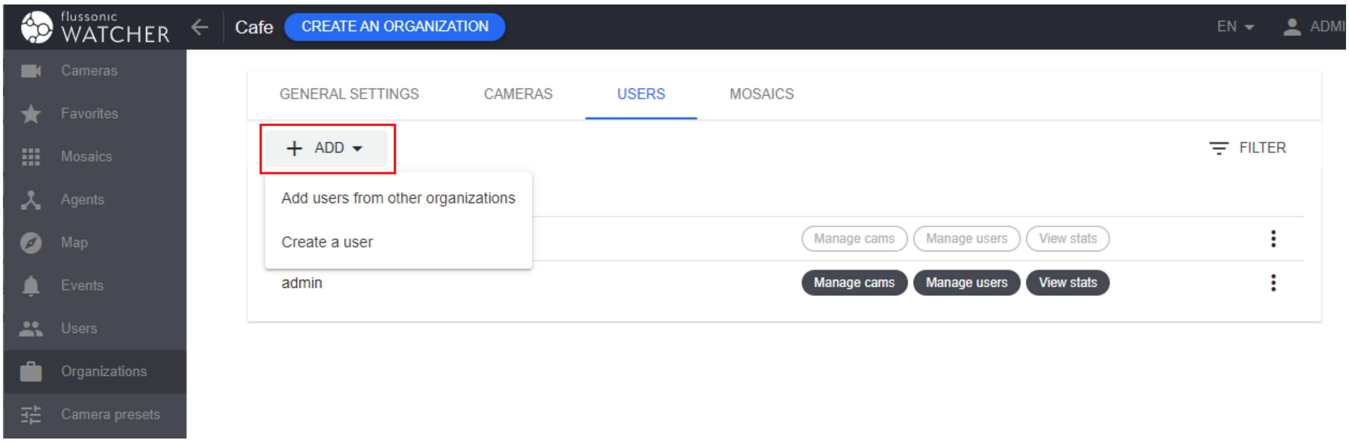
Adding users to an Organization

After you have added an Organization to Watcher, you will probably need to add users who will have access to cameras of this Organization. One user can be added to several Organizations.

To add a user to an Organization:

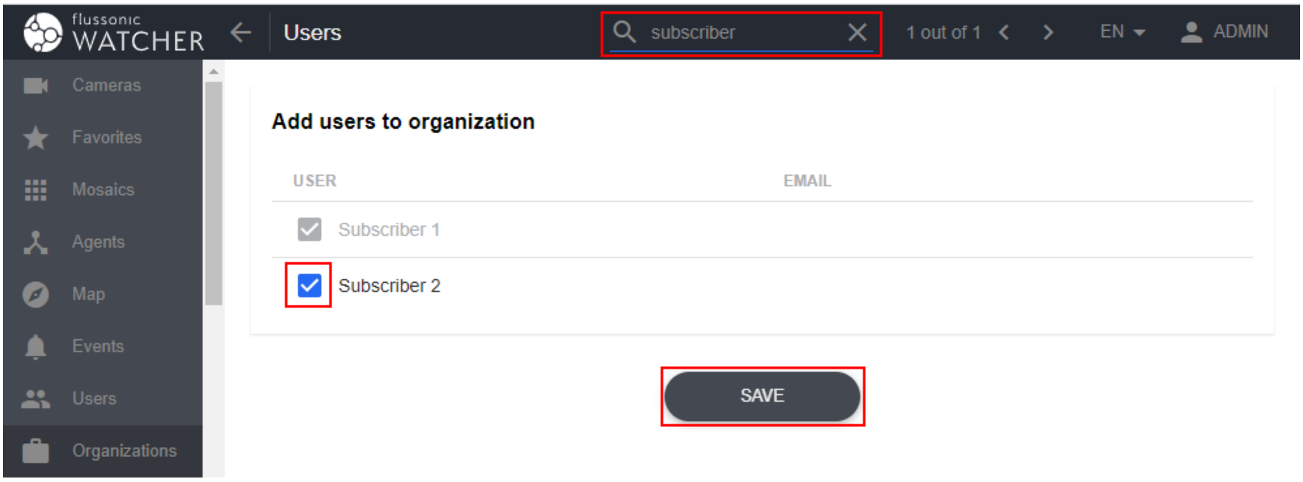
1. Go to **Organizations**.
2. In the list of Organizations, find the Organization where you are going to add a user (or create one) and click the counter in the **Users** column.

   You can also click the name of the Organization and go to the **USERS** tab in the Organization profile.
3. The page that opens shows the list of all users of this Organization. Click **ADD**.


flussonic
WATCHER
Cafe
CREATE AN ORGANIZATION
EN
ADMI
Cameras
Favorites
Mosaics
Agents
Map
Events
Users
Organizations
Camera presets
GENERAL SETTINGS
CAMERAS
USERS
MOSAICS
ADD
FILTER
Add users from other organizations
Create a user
Manage cams
Manage users
View stats
admin
Manage cams
Manage users
View stats

- Select **Add users from other organizations** to add existing user(s). The list of Watcher users open with users already added to this Organization highlighted in grey.

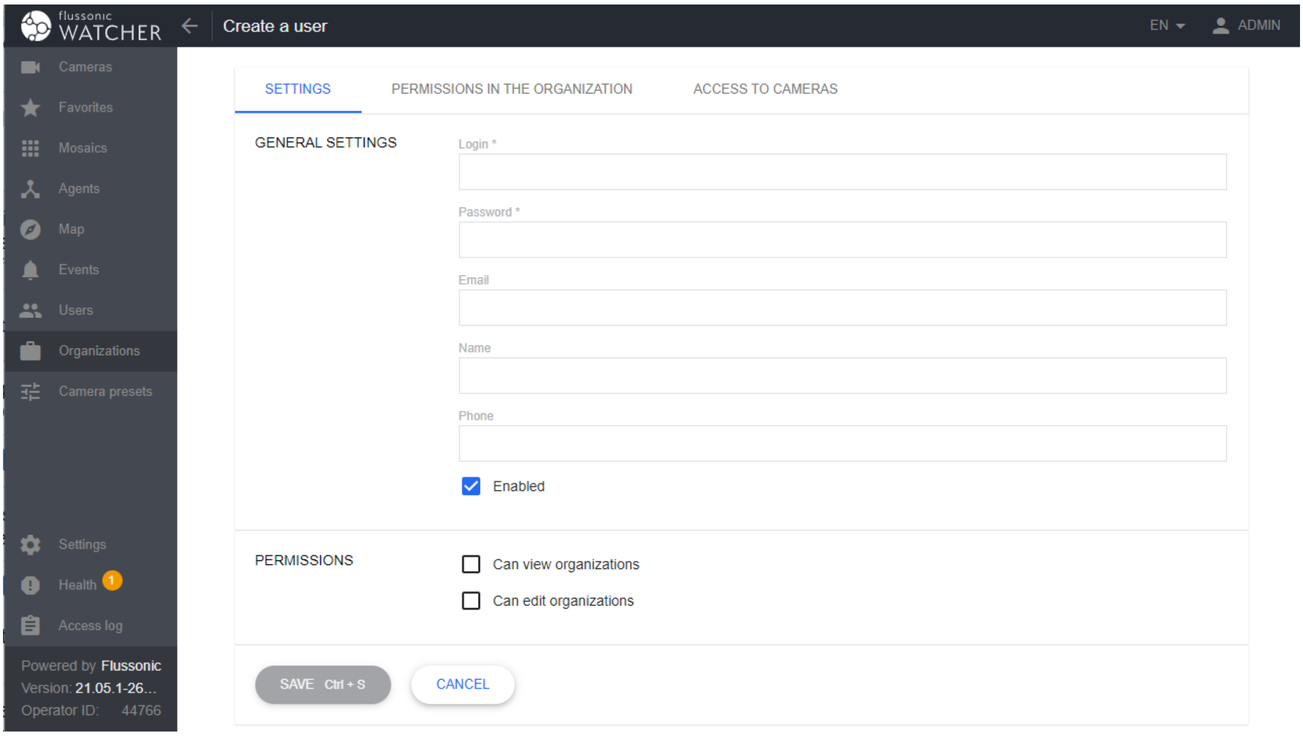
  Search for users by their login using the field in the top of the page. Check the users you want to add and click **Save**.



Add users to organization

USER	EMAIL
☑ Subscriber 1	
☑ Subscriber 2	

- Select **Create a user** to create a new user within this Organization, then fill in user data and save changes.



Powered by Flussonic
Version: 21.05.1-26...
Operator ID: 44766

GENERAL SETTINGS

Login *

Password *

Email

Name

Phone

☑ Enabled

PERMISSIONS

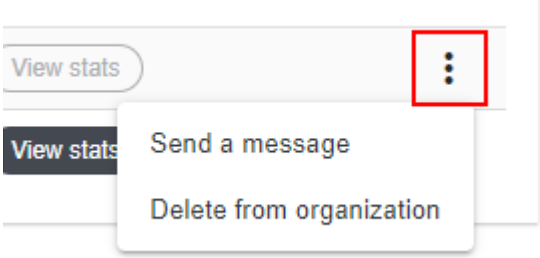
□ Can view organizations

□ Can edit organizations

Please note that you need to fill in fewer fields when creating a user this way compared to creating a user from scratch, for example, you do not need to select an organization, because it is already known, and there is no way to make the user Watcher Administrator or assign read-only rights. To set these parameters, you need to go to the user profile through the **Users** menu.

Deleting users from an Organization

If you want to delete a user from the Organization, go to **USERS** tab in the Organization profile and select the corresponding item in the three-dot menu on the right in the user row:

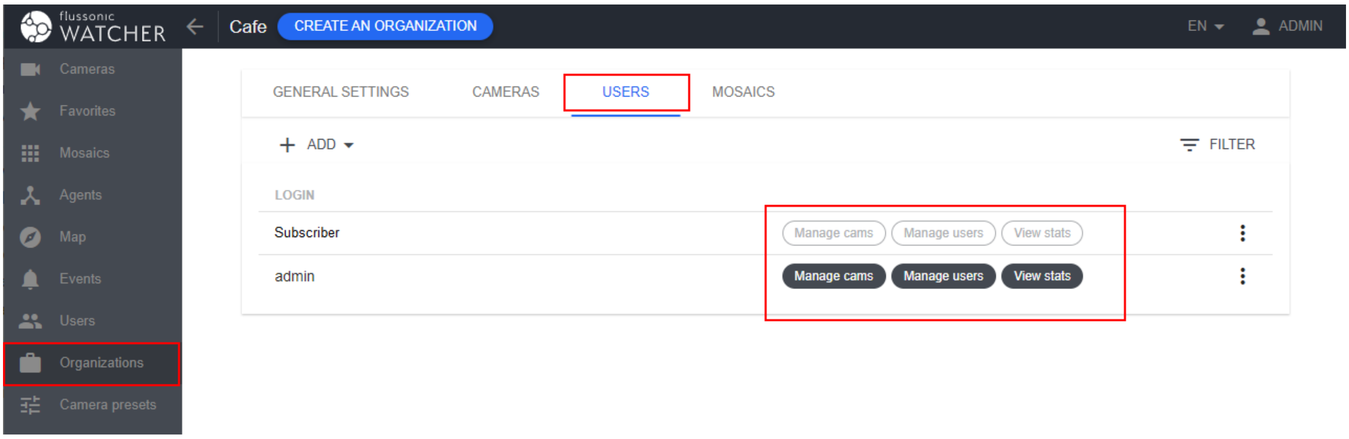


Granting a user rights to manage Organizations

One Organization can have a number of users and any user can belong to several Organizations. You can give each user different permissions to manage each Organization.

To change a user's rights to manage an Organization:

1. Go to **Organizations** and click in the column **Users** next to the Organization where you want to grant users rights to manage the Organization.



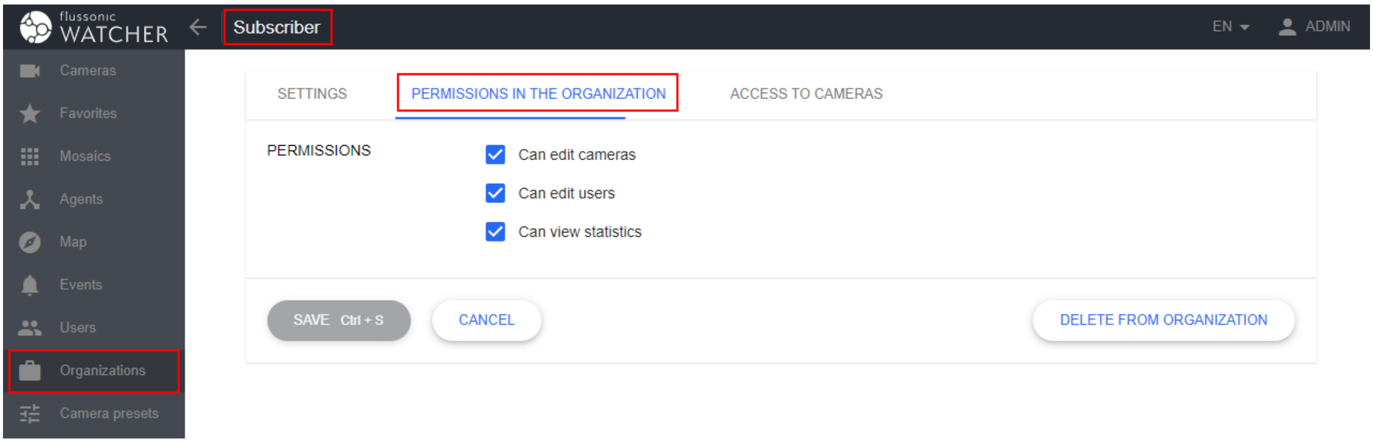
> **Note**
>
> Use **FILTER** to search for users by their login.

2. In the list of users that opens, click next to each user the permissions that you are giving them:

   - **Can edit cameras**: the user can add and edit cameras
   - **Can edit users**: the user can add and edit other users
   - **Can view statistics**: the user can view Watcher resources consumed by the Organization.

> **Note**
>
> The user who is Organization's administrator (owner) has all rights within the Organization. In this case you won't be able to save changes with some of the boxes on this tab unchecked.

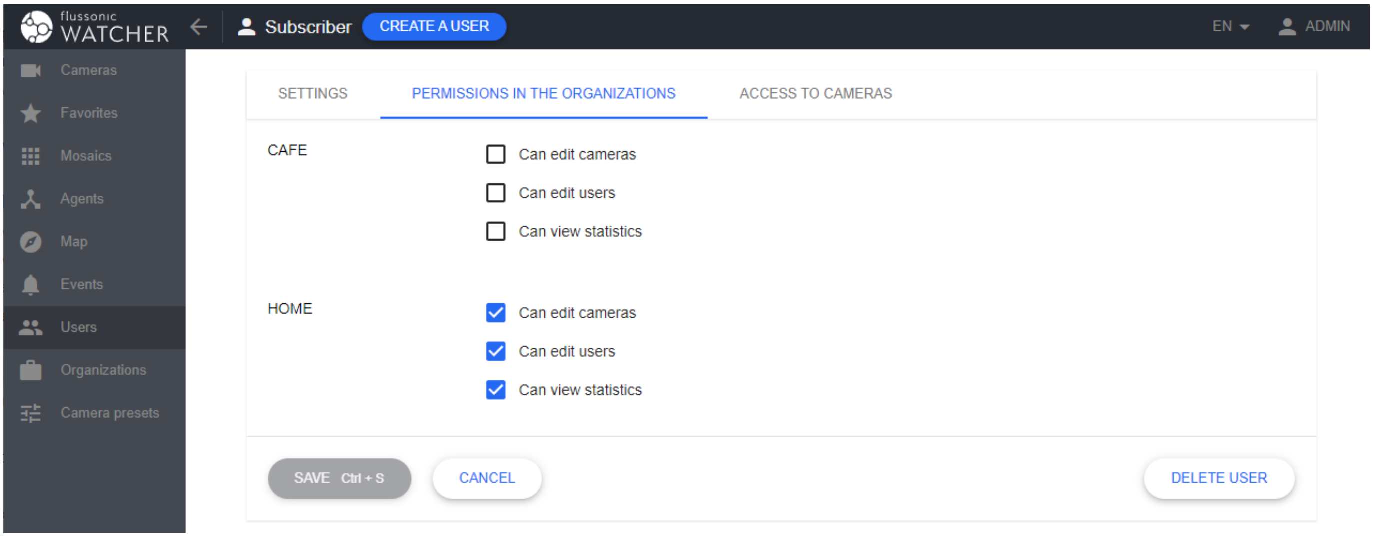
Another way to edit the user permissions to manage the Organization is in the user settings opened from **Organizations** — **Users** on the **Permissions in the Organization** tab.



PERMISSIONS

- ☑ Can edit cameras
- ☑ Can edit users
- ☑ Can view statistics

If a user is added to several organizations, you can edit access rights to several organizations by opening the user profile from the **Users** menu. In this case, the user rights in all owned organizations are available for editing on the **Rights in the organizations** tab:



CAFE

- □ Can edit cameras
- □ Can edit users
- □ Can view statistics

HOME

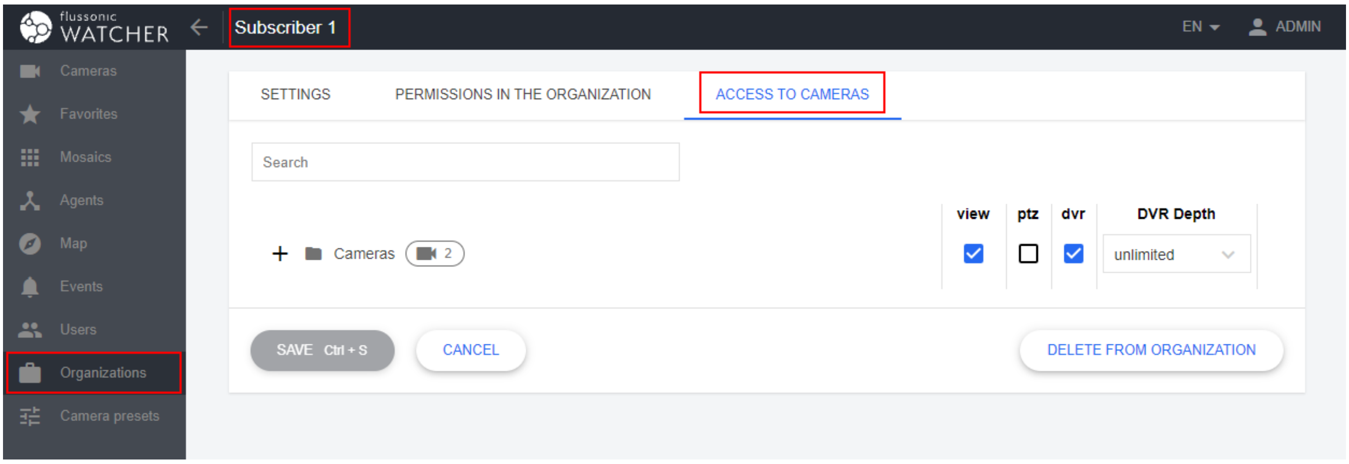
- ☑ Can edit cameras
- ☑ Can edit users
- ☑ Can view statistics

Granting a user access to cameras

Users can have various permissions to access video received from cameras.

To grant a user access to cameras:

1. Go to **Organizations** and click the number in the **Users** column next to the Organization where you want to modify user permissions.
2. In the list of users that opens, click the user whose permissions you want to modify.
3. In the user settings, go to the tab **Access to cameras**.



	view	ptz	dvr	DVR Depth
+ Cameras 2	☑	□	☑	unlimited

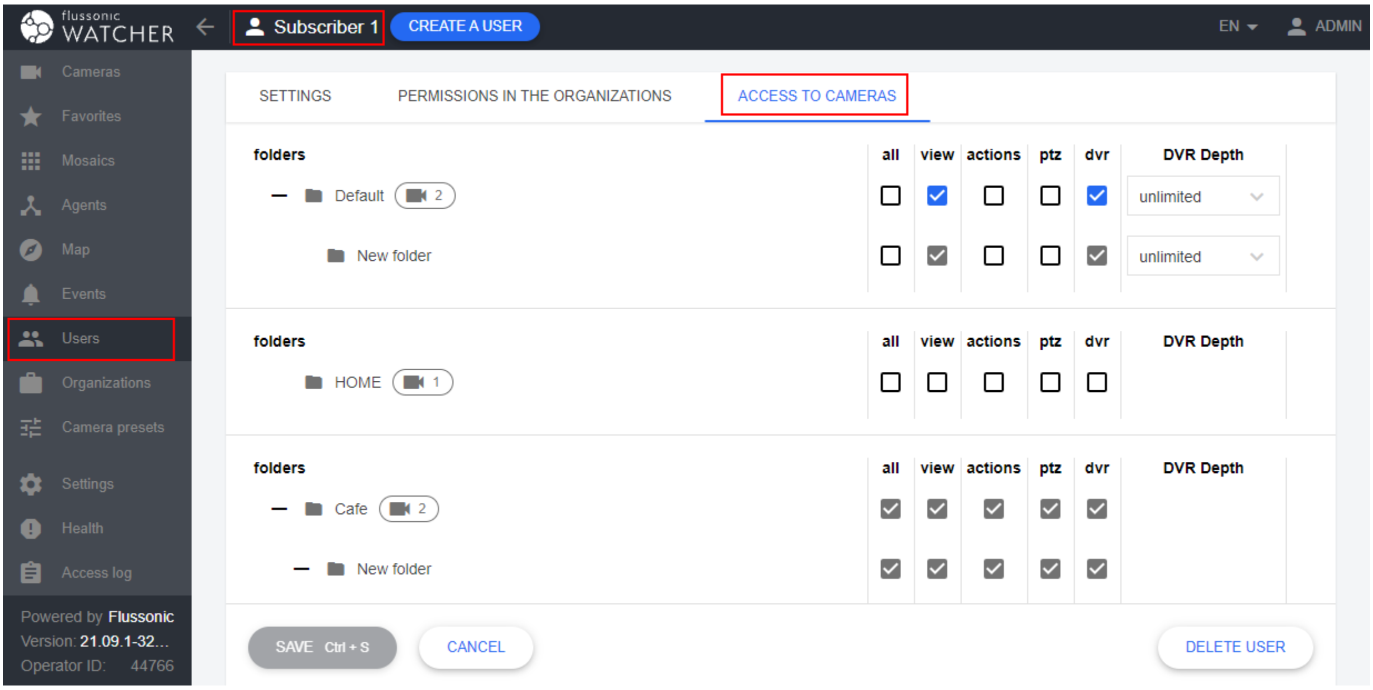
If the **Access to cameras** tab in inactive, the user is not allowed to manage cameras. To allow managing cameras, grant permissions as described above.

> **Note**
>
> The user who is the owner (administrator) of the Organization has all rights to the cameras in this Organization. In this case the boxes cannot be unchecked and the drop-down list for archive depth adjustment is not displayed.

4. Select folders containing cameras to which you grant this user access to. The selected cameras will be available to the user in the Dashboard.

   Access categories are described below.

It is also convenient to edit the user's rights to cameras in all organizations to which the user is added by opening the profile from the **Users** page and selecting the **Access to cameras** tab:



Powered by Flussonic
Version: 21.09.1-32...
Operator ID: 44766

folders	all	view	actions	ptz	dvr	DVR Depth
— Default 2	□	☑	□	□	☑	unlimited
New folder	□	☑	□	□	☑	unlimited

folders	all	view	actions	ptz	dvr	DVR Depth
HOME 1	□	□	□	□	□	

folders	all	view	actions	ptz	dvr	DVR Depth
— Cafe 2	☑	☑	☑	☑	☑	
— New folder	☑	☑	☑	☑	☑	

CATEGORIES OF ACCESS TO FOLDERS WITH CAMERAS

- **Access to viewing cameras** — the user can view live video received from IP cameras of the Organization.
- **Access to DVR archive** — the user can view recorded video in DVR archives.
- **Access to PTZ** — the user can control cameras via PTZ.
- **Access to actions** (displayed in user settings only) — the user can add, edit or execute actions on the Cameras page.
- **Custom DVR depth** (displayed if the user is not Organization's administrator and has access to DVR) — the user has custom permissions to access the archive of cameras within the folder. For example, if the camera has 1-week-deep archive, you can provide your subscribers with access to just last day recordings and charge for "unlimited" access permissions.

> **Note**
>
> Granting a user access to a folder means that all its subfolders will also be available to this user.

> **Warning**
>
> You can give access only to an entire folder, not to individual cameras in the folder. To give access to only one camera, add this camera to a separate folder and give the user access to that folder.

PTZ works inside a player in Watcher UI only. In the embed player being accessible via the "Shared URL" the control elements for PTZ shall not be shown.

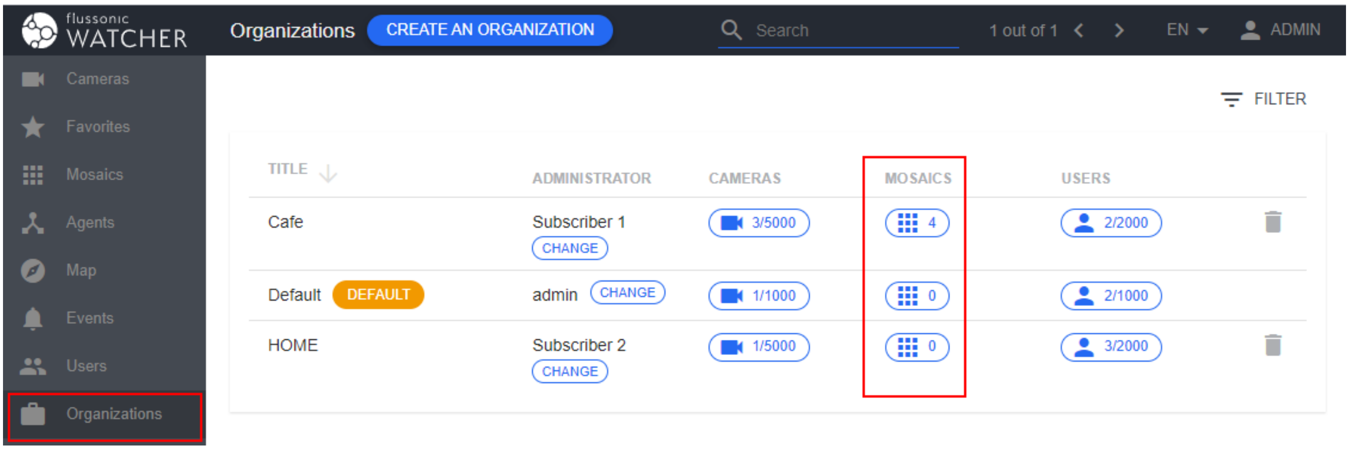
4.3.10 Creating a client mosaic

A **Mosaic** is a page with several simultaneously viewed cameras. You can gather up to 64 cameras on a mosaic. The mosaic usually includes cameras from the same organization as described on this page. However, the Dashboard mode allows adding cameras from several organizations to the mosaic if it does not contradict with the user permissions.

Managing mosaics

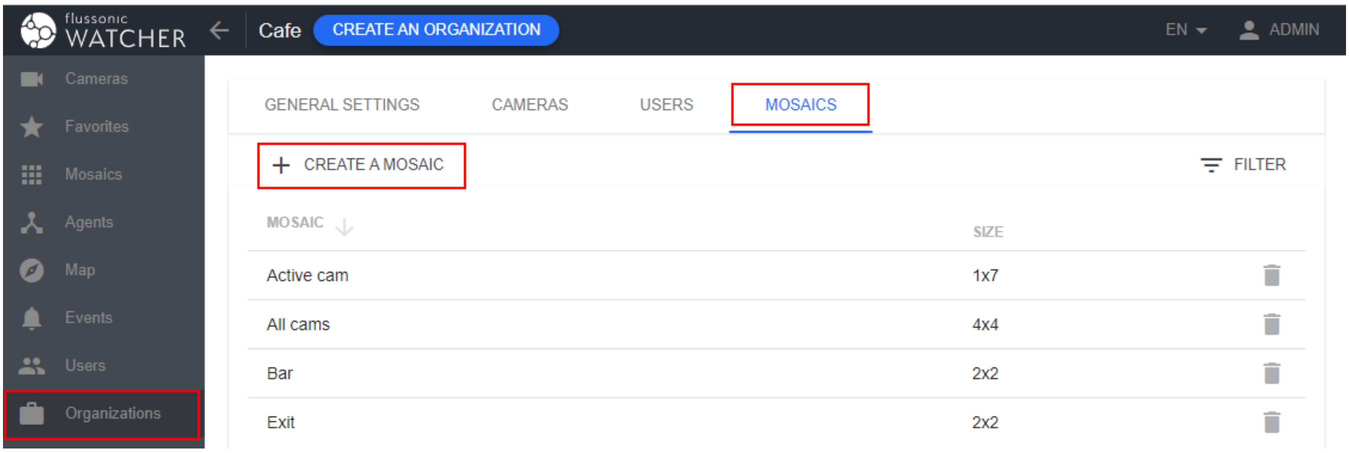
How to create a mosaic:

1. Log in to Flussonic Watcher as an Organization's administrator (owner).
2. Go to **Organizations** and click the mosaic counter in the row of the Organization:



TITLE	ADMINISTRATOR	CAMERAS	MOSAICS	USERS
Cafe	Subscriber 1 CHANGE	3/5000	4	2/2000
Default DEFAULT	admin CHANGE	1/1000	0	2/1000
HOME	Subscriber 2 CHANGE	1/5000	0	3/2000

3. Click **Create mosaic** or select an existing mosaic to edit.

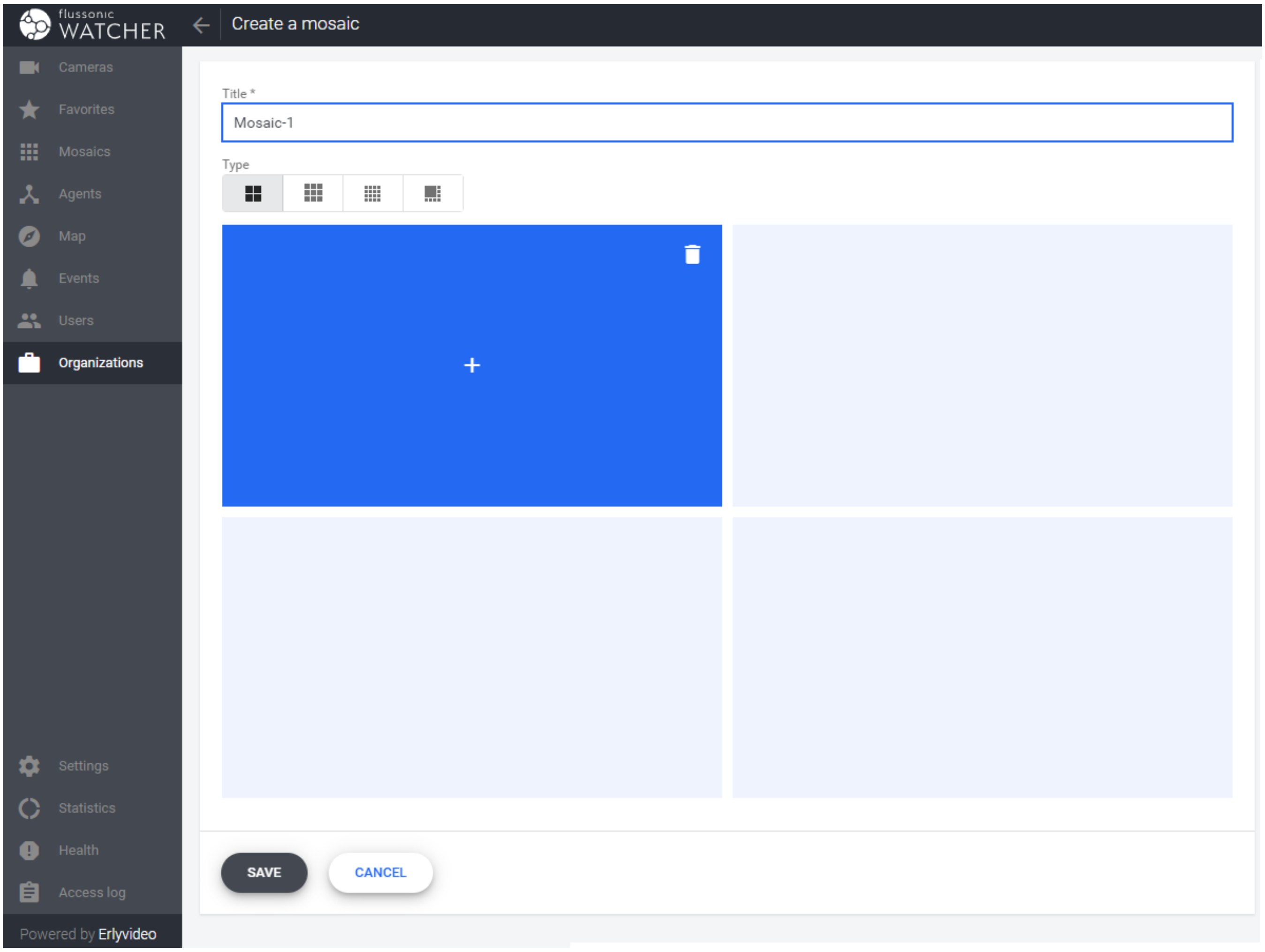


MOSAIC	SIZE
Active cam	1x7
All cams	4x4
Bar	2x2
Exit	2x2

Note

You can also delete mosaics using the trash bin icon.

4. In the window that opens, enter the mosaic title and select the dimensions of the mosaic (the number of cameras in it).

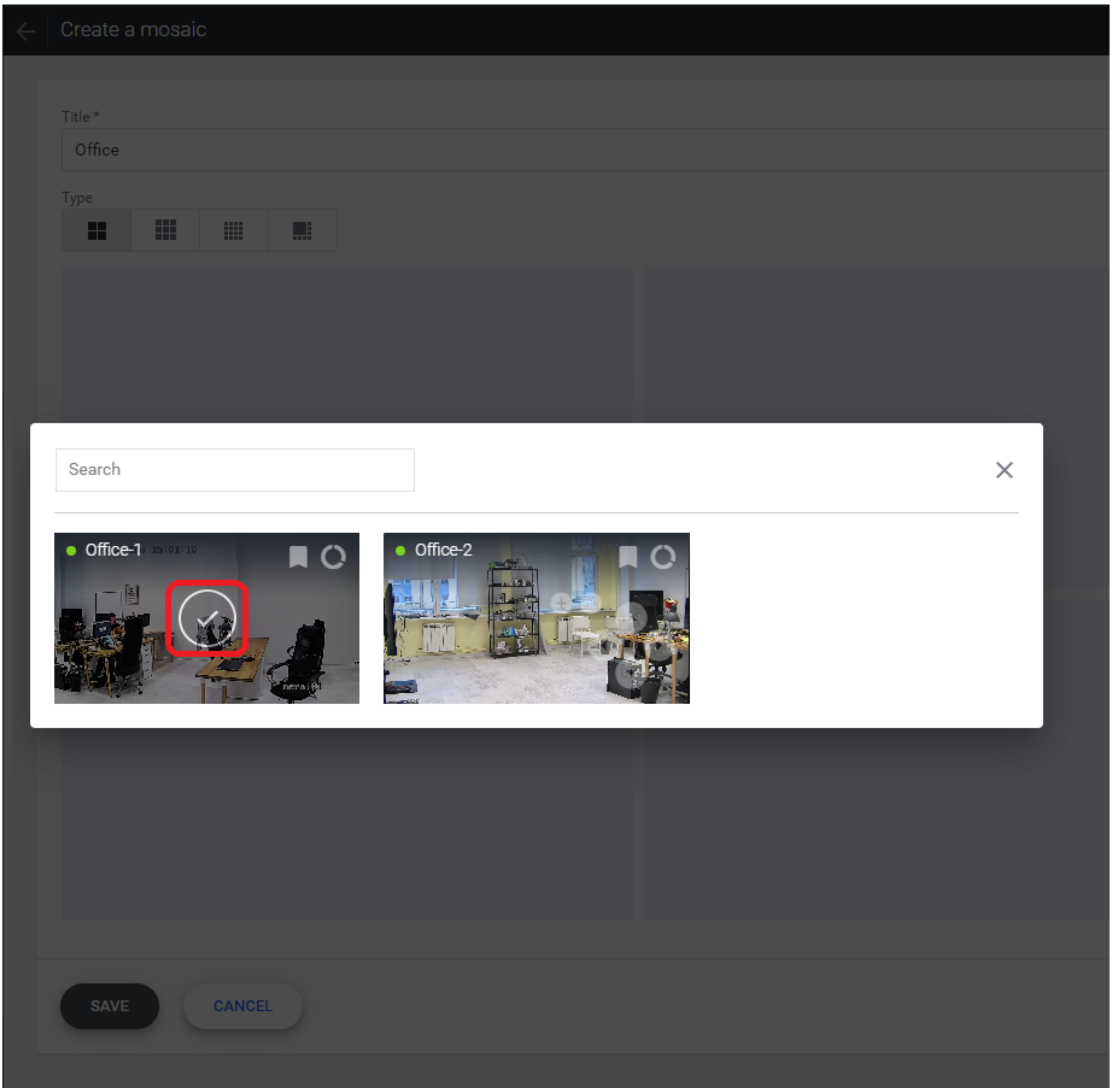


Title *

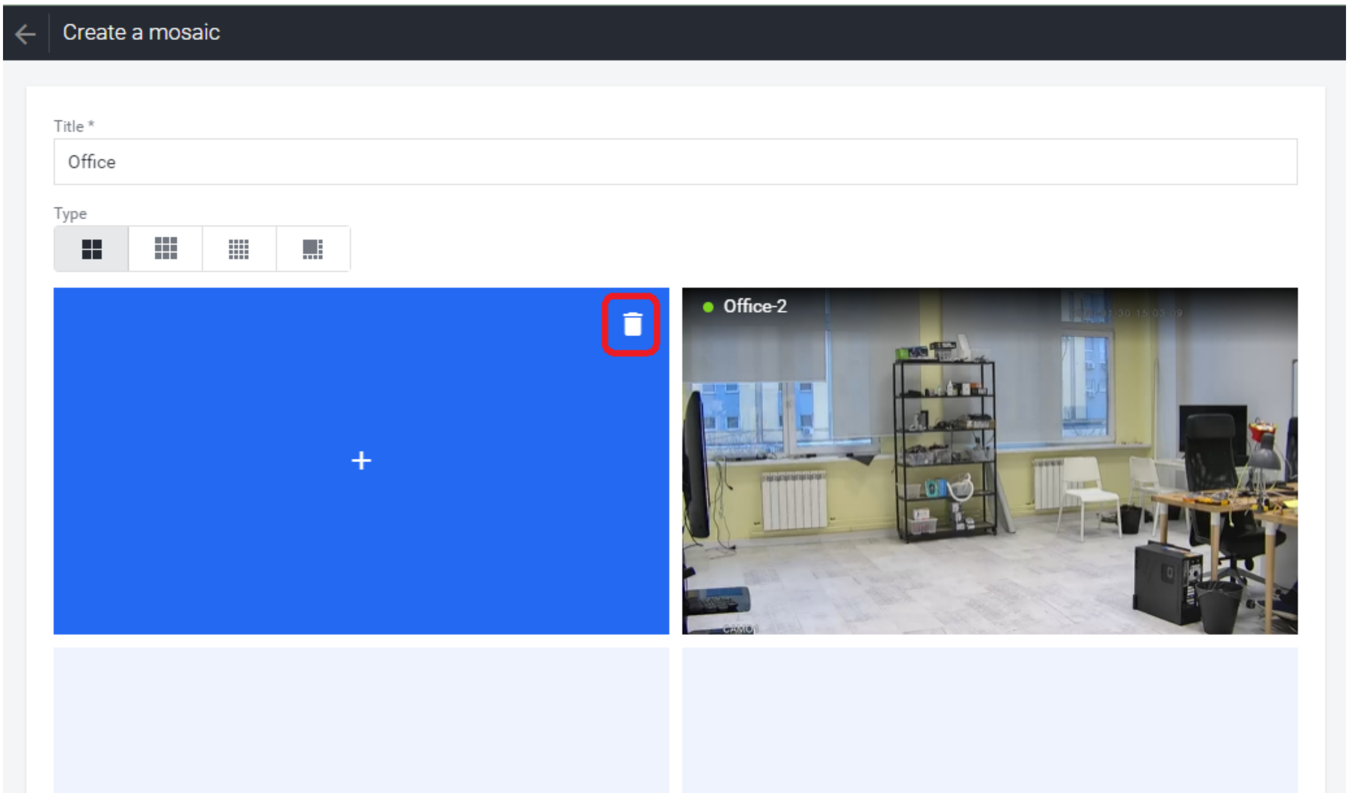
Mosaic-1

Type

5. Click the mosaic tiles to select cameras.



6. To delete a camera from the mosaic, click the recycle bin icon in the upper right corner of the corresponding tile in the mosaic.



Create a mosaic

Title * Office

Type

Using mosaics

You can operate the mosaics in the following ways:

- In the **Mosaics** section: select a mosaic to view.
- In the **Cameras** section: use the Dashboard to view, edit, copy or delete mosaics.
- In the **Organizations** section: edit or delete mosaics in the same way as you create them.

4.3.11 Sending a Camera stream to an External Service

Sending a stream to an external service using the push model is used for:

- Sending video to social networks (YouTube, Facebook, Telegram)
- Sending video to IPTV network.
- Other integrations when you know the protocol the external system is ready to accept video.

Watcher allows you to send a stream from an IP camera in any way available in Media Server. This means you can specify either an RTMP link or a multicast group address to send MPEG-TS.

We provide this feature only to server administrators, as each added link consumes server resources and only the system administrator can know the network addressing when configuring advanced scenarios.

Push Configuration in the Admin Menu

- Open the **Admin** menu, go to the **Cameras** page.
- Select the desired camera.
- Go to the **Pushes** tab.
- Click the **Add** icon and enter the external service URL.

Social networks usually provide separate details:

- **Server URL**: rtmp://a.rtmp.youtube.com/live2
- **Stream key**: auyj-u5ww-jm2c-jdv6-9q5u *(this is an example, do not copy it)*

Combine them into one string to get `{server_url}+{stream_key}`, for example: `rtmp://a.rtmp.youtube.com/live2/auyj-u5ww-jm2c-jdv6-9q5u`.

If necessary, use `/` as a separator.

This will be the URL you need to insert into Watcher.

If everything is set up correctly, within 5-10 seconds you will see the video on the external service.

Final words

Watcher allows sending a stream from a camera not only to social networks but also to IPTV networks, to arbitrary URLs that accept publication, enabling administrators to solve various integration tasks of the video surveillance system.

4.4 Video analytics

4.4.1 Video analytics installation

We refer to video analytics as the process that, using computer vision algorithms, records episodes with a description of objects in the frame. To use it, you should install special software. This software will henceforth be referred to as the **analytics module**.

The analytics module requires a significant amount of computational resources, therefore we strongly recommend dedicating a separate server and utilizing hardware accelerators.

Note

For testing purposes, it is permissible to:

- run the analytics on the same server where the IP cameras are ingested and recorded
- run the analytics on a CPU

Analytics performs well on a CPU, but it does not leave free resources for other programs.

Requirements to the server for video analytics

- OS: x64 Ubuntu 20.04 or 22.04.
- GPU: NVIDIA with at least 6 GB VRAM.
- Processor: at least 4 cores.
- Memory: at least 8 GB RAM.

  The main characteristic when selecting a compatible graphics card is the **Compute Capability version: 6.1 or higher** is required. You can select a graphics card on the official NVIDIA website.

  See also Calculating video memory for analytics on GPU.

Installing video analytics

Run the commands:

```
apt update
apt install --no-install-recommends nvidia-driver-525
apt install flussonic-vision
```

After installing the packages, go to the Watcher UI and add new streamers with the analytics role as described further.

Adding analytics to Watcher

The analytics module consists of two components: Inference and Identification. One is needed for "capturing images" from the stream, and the other for their identification.

If you have only one analytics server, then install both components. If you have multiple servers, it is not mandatory but it is advisable to install both components on each server: this enhances the system's performance and fault tolerance.

ADD INFERENCE

1. Open the **Streamers** page.
2. Click **+** to add a streamer.
3. In the form, specify the server name. You can use the actual hostname, or any arbitrary string. Specify `inference1`, which will work for the first installation.
4. Choose the **Inference** role. Save.

5. Go to the streamer settings page and specify the server's `API_URL`: `http://secret@vision.example.com:9030`

Remember the name `inference1` and the API key `secret` as you'll need those in the last step.

ADDING IDENTIFICATION

The actions are similar to adding Inference, but you should specify a different role and port.

Note that the server name must be unique. If you installed two components, they will appear as separate streamers in Watcher.

1. Open the **Streamers** page.
2. Click **+** to add a streamer.
3. In the form, specify the server name. You can use the actual hostname, or any arbitrary string. Specify `identification1`, which will work for the first installation.
4. Choose the **Inference** role. Save.
5. Go to the streamer settings page and specify the server's `API_URL`: `http://secret@vision.example.com:9050`.

Remember the API key `secret` as you'll need it in the last step.

In this example, `secret` is the `API_KEY` and it is specified as part of the URL. Do not enter it in the Cluster key field.

EXAMPLE CONFIGURATION FOR A TEST ENVIRONMENT

- Hostname: `streamer.lab`. API_URL: `http://streamer.lab`. Role: `Streamer`.
- Hostname: `inference1`. API_URL: `http://secret@vision.lab:9030`. Role: `Inference`.
- Hostname: `id1`. API_URL: `http://secret@vision.lab:9050`. Role: `Identification`.

Video analytics settings

On the analytics module server, set the config in `/etc/vision/vision-inference.conf` and `/etc/vision/vision-identification.conf` files.

INFERENCE SETTINGS

Open the configuration file `/etc/vision/vision-inference.conf` and specify the following:

1. `API_KEY` that you used in the previous steps.
2. `HW`. By default, `HW=cuda` is specified in the config. This means that the module will try to work with the NVIDIA video card. Specify `HW=cpu` if you don't have a video card yet.
3. In `CONFIG_EXTERNAL`, specify the address of the server with Watcher so that the analytics receives information about which cameras to analyze.

In the below example, you should change the following data:

- `CENTRAL_KEY` is the API key that can be found in `/etc/central/central.conf`.
- `watcher.lab` should be your real Watcher hostname.
- `NAME` is the server's name. In the example above we used `inference1`.

```
HTTP_PORT=9030
API_KEY=secret
HW=cuda
CONFIG_EXTERNAL=http://CENTRAL_KEY@watcher.lab/central/api/v3/streamers/NAME/streams
```

`HTTP_PORT` should be the same as you used in Watcher for the streamer with `inference` role

IDENTIFICATION SETTINGS

Open the configuration file `/etc/vision/vision-identification.conf` and specify the following:

1. `API_KEY` that you used in the previous steps.
2. In `CENTRAL_URL`, specify the address of the server with Watcher so that the analytics receives information about which cameras to analyze.

In the below example, you should change the following data:

- `CENTRAL_KEY` is the API key that can be found in `/etc/central/central.conf`.

```
HTTP_PORT=9050
API_KEY=secret
CENTRAL_URL=http://CENTRAL_KEY@watcher.lab/central/api/v3
```

`HTTP_PORT` should be the same as you used in Watcher for the streamer with `identification` role.

STARTUP AND TESTING

After this, you can run both components:

```
systemctl start vision-inference vision-identification
```

If all the settings are correct, you will see green indicators on the **Streamers** page in the UI.

4.4.2 Video analytics upgrade to actual version

UPDATING VIDEOANALYTICS

Run the following commands to update videoanalytics: echo "deb http://apt.flussonic.com/repo binary/" > /etc/apt/sources.list.d/flussonic.list; apt update && apt install flussonic-vision

INSTALLING A ROLLING UPDATE (MASTER) VIDEOANALYTICS PACKAGE

Change the repository to the one with the rolling updates and install videoanalytics: echo "deb http://apt.flussonic.com/repo master/" > /etc/apt/sources.list.d/flussonic.list; apt update && apt install flussonic-vision

ROLLING VIDEOANALYTICS BACK TO THE PREVIOUS VERSION

Find out the available previous versions using `apt show`: apt show flussonic-vision -a | grep Version

Install regarding the received versions: apt install flussonic-vision=24.05

4.4.3 License Plate Detection Events

Flussonic Watcher can detect license plates and recognize license plates on the video transmitted by an IP camera including non-standard formats. This functionality is known as ANPR (automatic number plate recognition) or LPR (license plate recognition).

Flussonic Watcher does the following:

- Creates events of license plate detection. Video comes from IP cameras to a streamer (in cluster mode) or to the managing server (in single mode), where the license plate recognition takes place.
- Provides the Watcher UI for viewing plate detection events. You can view registered events and watch the recorded video of each event.
- Provides the API for integration with external services.

License plate recognition is available for the following countries:

- All EU (European Union) countries
- Moldova
- Russia
- Abkhazia
- Ukraine
- Armenia
- Kazakhstan
- Bulgaria
- Indonesia

To start detecting license plates:

1. Prepare hardware and software for the Flussonic server that will carry out license plate recognition — see Video analytics module.
2. Check the video image parameters for compliance with the recommendations (see below).
3. Turn on and configure the license plates recognition.

On this page:

- Recommended video image parameters for LPR
- Setting up LPR
- Viewing LPR events in Watcher

Recommended video image parameters for LPR

A general requirement for a video image for LPR is the ability to read the license plate with human eyes. In other words, if you do not perceive the license plate in the video, then Watcher will not be able to recognize it either.

Stable LPR is guaranteed with the following video image characteristics:

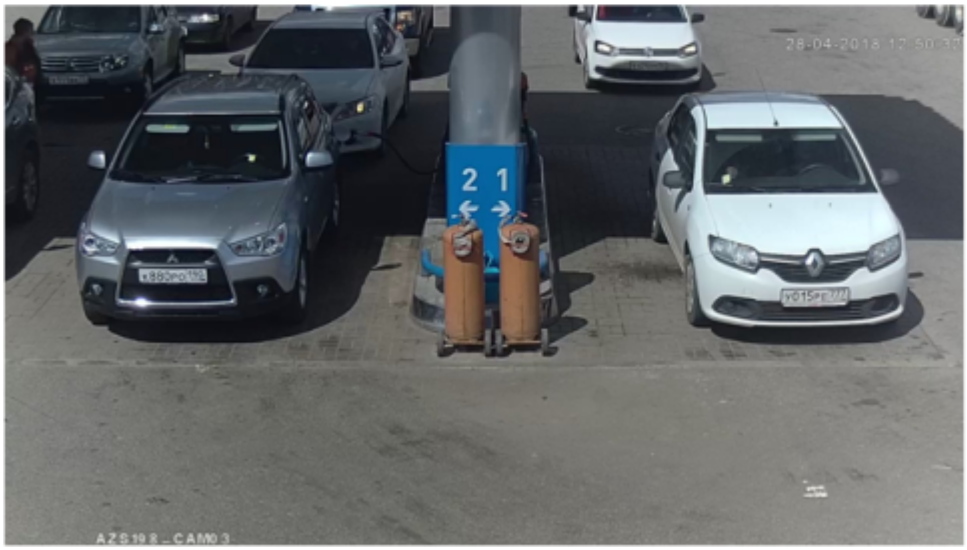
1. At least HD resolution (1280 x 720). If the frame has larger resolution, it will be reduced to 720p before feeding to the analytics module. Also, if camera has several streams, the closest to 720p will be used for analytics purposes.
2. The size of the license plate in the image at least 100 x 20 pixels at 720p. If the frame has larger resolution, the LP size should be proportionally larger.
3. Good illumination.
4. Colored image is better than black-and-white.
5. Any camera view angle.

However, if your video image does not meet these guidelines or even if your country is not supported yet, you can check to see if recognition will work. If the LPR quality turns out to be unsatisfactory, we may consider the module modification to suit your conditions. Please contact our technical support team by following these instructions. Technical support team member will request the necessary information and will let you know if the improvements are possible and when we will implement them.

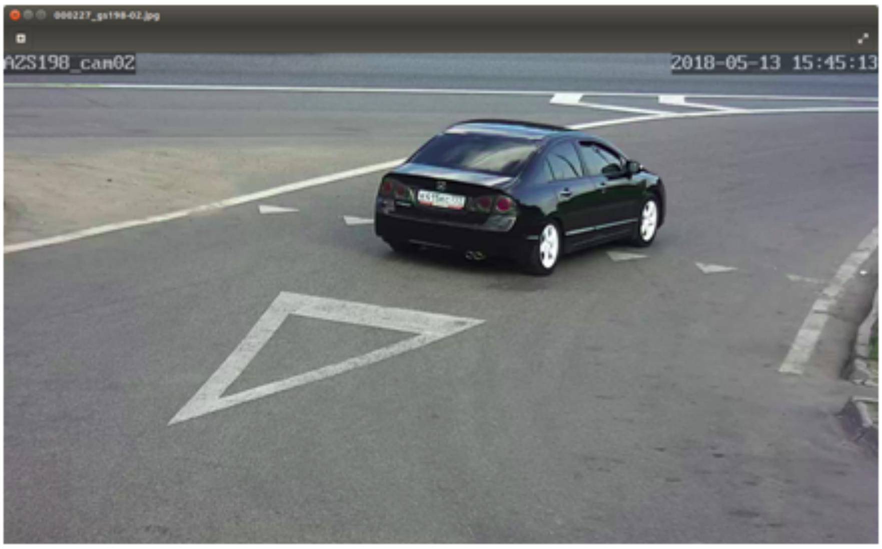
Please note that if your country is already on the list, then access to video stream(s) where the recognition quality does not meet expectations is enough. If you want us to support the license plates recognition for a new country, then most likely you will be asked to provide access to two types of streams:

- The camera is aimed at the license plate at a right angle (90°)
- The camera is aimed at the license plate at an arbitrary angle

Example right angle



Example arbitrary angle

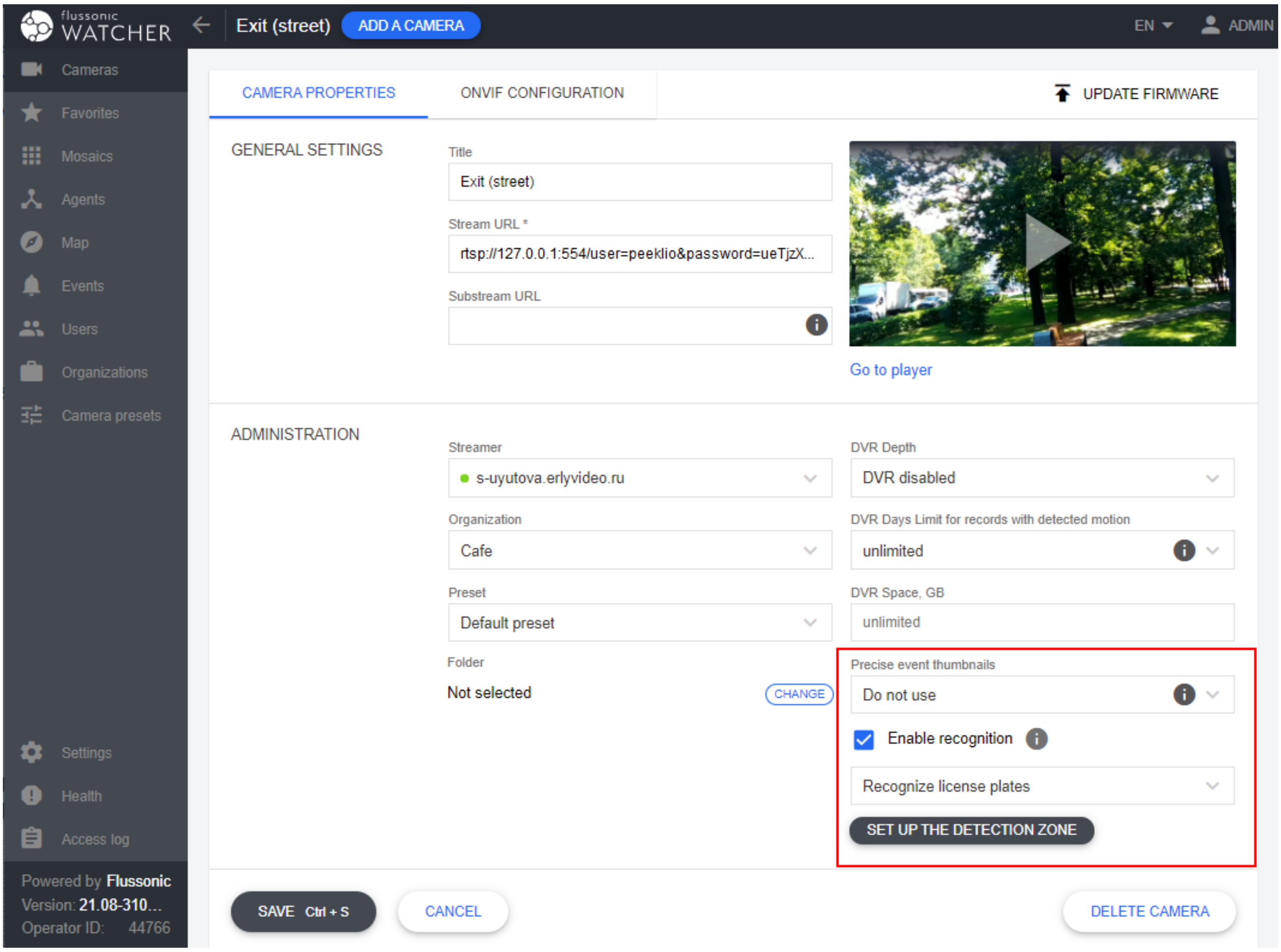


Technical support team member will provide a detailed info at your request.

Setting up LPR

To turn on license plate detection and recognition for a camera:

1. In the Watcher UI, go to **Cameras**. Find the camera in the list and open its settings by clicking the icon in the upper right corner of the player.
2. Check the **Enable recognition** box.



GENERAL SETTINGS

Title: Exit (street)

Stream URL *: rtsp://127.0.0.1:554/user=peeklio&password=ueTjzX...

Substream URL:

ADMINISTRATION

Streamer: s-uyutova.erlyvideo.ru

Organization: Cafe

Preset: Default preset

Folder: Not selected

DVR Depth: DVR disabled

DVR Days Limit for records with detected motion: unlimited

DVR Space, GB: unlimited

Precise event thumbnails: Do not use

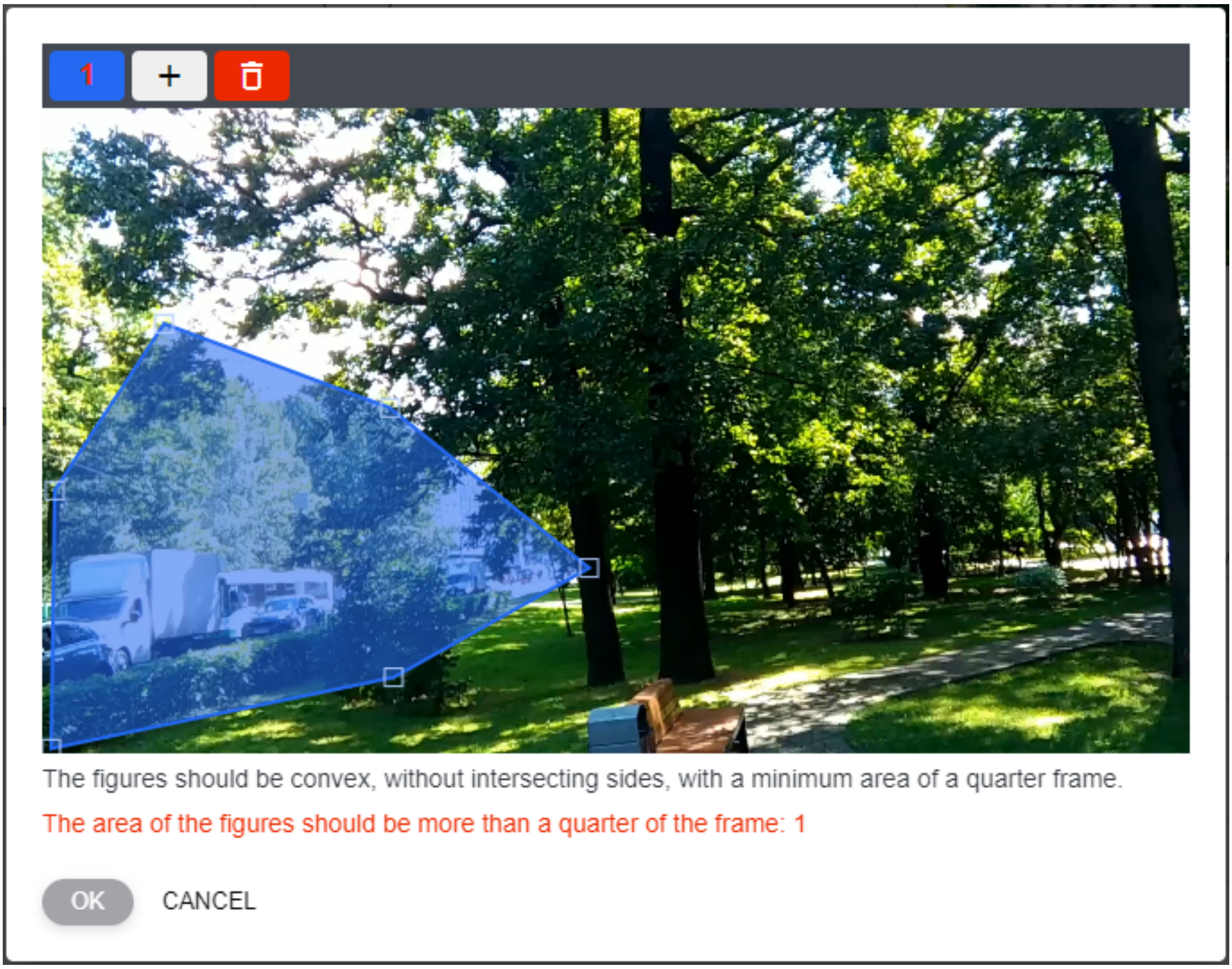
☑ Enable recognition

Recognize license plates

3. Select **Recognize license plates** in the drop-down list that appears when you check the box.
4. Select the period to store precise thumbnails for, if necessary. Please refer to Camera settings for details on precise thumbnails.

> **Note**
>
> The recognition settings may not be available if a non-adjustable preset is selected. In this case, select or create the preset with the settings you need.

5. By default, the recognition system searches for license plates over all the camera field of view. You can select specific polygonal area(s) to detect license plates in by clicking **Set up the detection zone**. This settings may help you to reduce false detections. The areas are set in a dialog:



The figures should be convex, without intersecting sides, with a minimum area of a quarter frame.

The area of the figures should be more than a quarter of the frame: 1

There are controls in the dialog header: buttons for selecting already created areas, as well as buttons for creating and deleting areas.

You can drag-and-drop the created area to the desired position on the frame and move the vertexes. If you want to delete an area, select it and click the trash can icon. Click **OK** when all areas are set.

Note

Make sure that your areas are convex and do not intersect each other. Any area should cover at least quarter of the frame. You will see a warning if any of the requirements is not fulfilled while the **OK** button will be disabled until you make fixes according to the remarks.

Note

Areas are set individually for each camera, even if you select a non-configurable preset.

6. Save the camera settings.

Now Flussonic Watcher will recognize license plates that appear in the frame of this camera, and mark the time when the vehicle entered and left the scene.

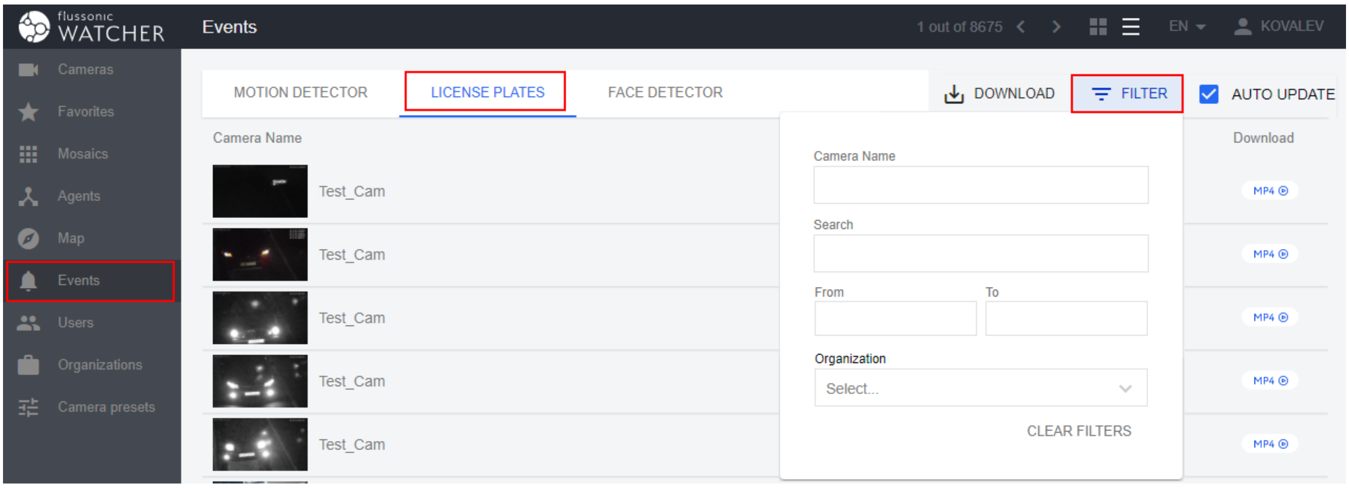
Viewing LPR events in Watcher

Flussonic Watcher creates events of two types:

- enter – a license plate appeared in the field of view of a camera
- leave – a license plate left the field of view.

To see detected license plates for a camera:

1. In the Watcher UI, go to **Events** - **License plates**. The list of all events opens.
2. You can use filter to find specific events:



Camera Name	Download
Test_Cam	MP4
Test_Cam	MP4
Test_Cam	MP4
Test_Cam	MP4
Test_Cam	MP4

Camera Name

Search

From To

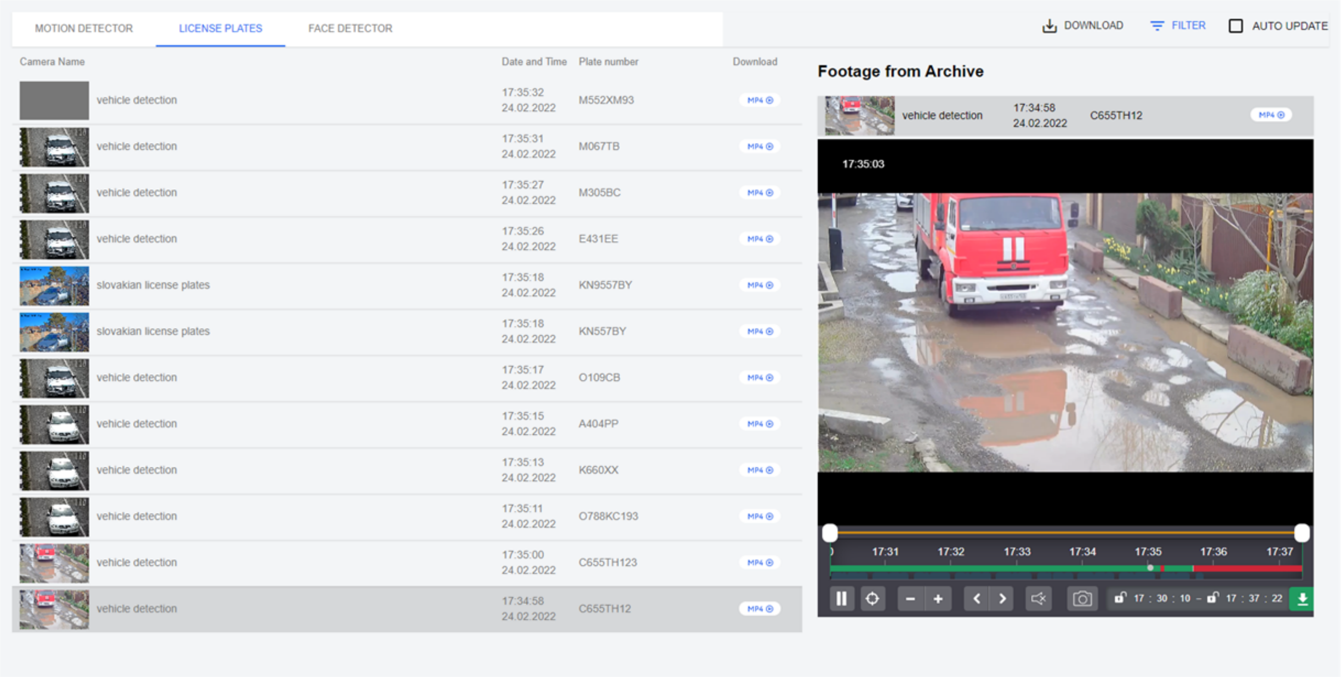
Organization

Select...

CLEAR FILTERS

- In **Camera**, enter the camera name.
- In **From** and **To**, select the date and time of beginning and end of the period when events were detected.
- In **Search**, type the license plate text (numbers and letters).
- From the **Organization** list, select the Organization that includes cameras events from which you want to see.
- To delete the specified search criteria, click **Clear Filters**.

3. The list of events is filtered as you enter search parameters.
4. Click the event to view its recording. The player will appear on the right, playback position will be set to the recording. You can control playback just like in the usual player.



Camera Name	Date and Time	Plate number	Download
vehicle detection	17:35:32 24.02.2022	M552XM93	MP4
vehicle detection	17:35:31 24.02.2022	M067TB	MP4
vehicle detection	17:35:27 24.02.2022	M305BC	MP4
vehicle detection	17:35:26 24.02.2022	E431EE	MP4
slovakian license plates	17:35:18 24.02.2022	KN9557BY	MP4
slovakian license plates	17:35:18 24.02.2022	KN557BY	MP4
vehicle detection	17:35:17 24.02.2022	O109CB	MP4
vehicle detection	17:35:15 24.02.2022	A404PP	MP4
vehicle detection	17:35:13 24.02.2022	K660XX	MP4
vehicle detection	17:35:11 24.02.2022	O788KC193	MP4
vehicle detection	17:35:00 24.02.2022	C655TH123	MP4
vehicle detection	17:34:58 24.02.2022	C655TH12	MP4

Footage from Archive

vehicle detection 17:34:58 24.02.2022 C655TH12 MP4

Note

Use controls in the header to page events or switch between screenshot and list view.

See also Events.

4.4.4 Detecting vehicles without license plate

Watcher can detect vehicles without license plates. This function is a part of license plate recognition (LPR) module.

Recognizing vehicles without license plates requires additional resources. Please refer to Calculating resources for analytics for details on the resources consumption.

Recommended video image characteristics for recognizing vehicles without license plates

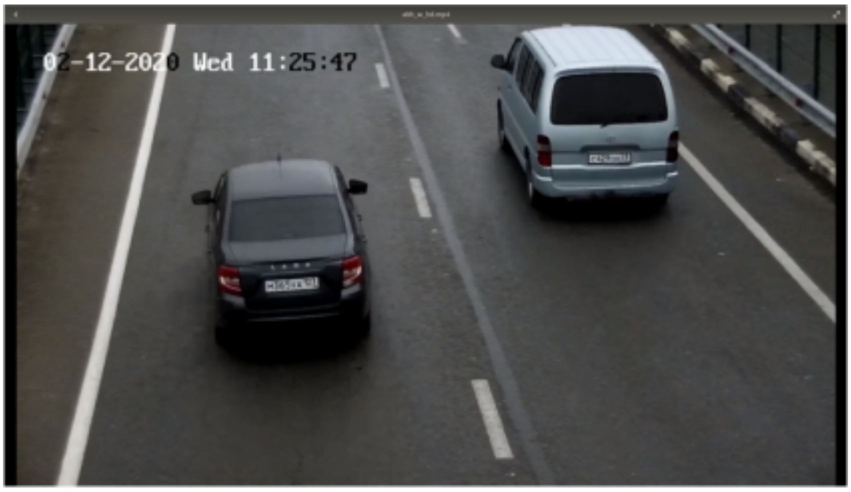
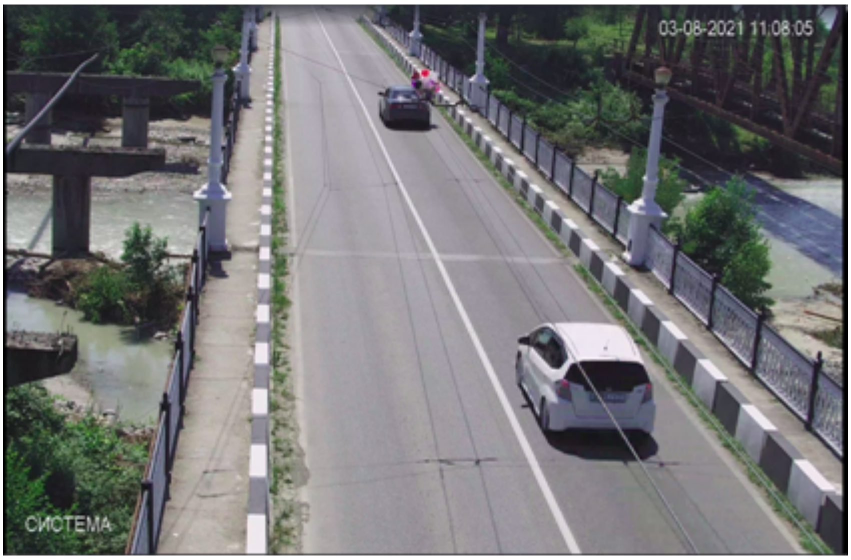
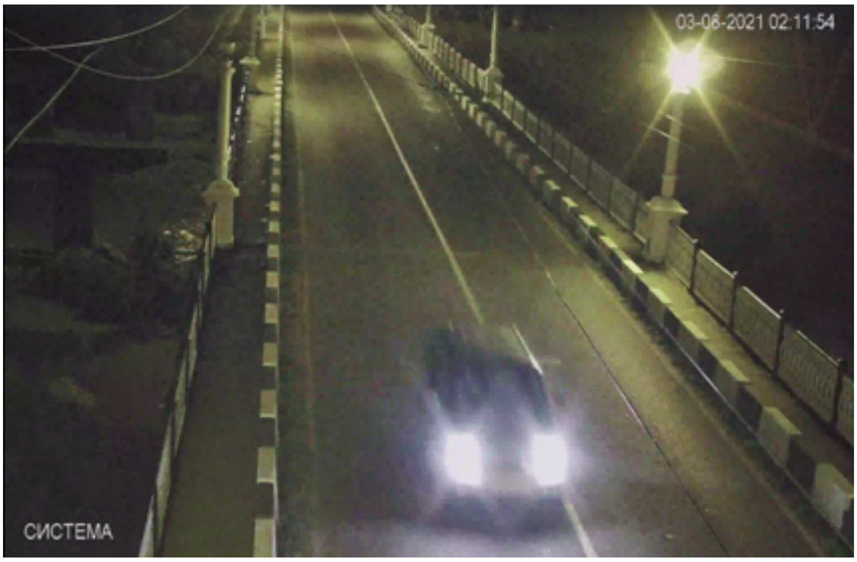
Camera requirements for detecting vehicles without license plates are slightly different from the requirements for recognizing license plates due to the module features.

The main recommendations are as follows:

- The license plate must be clearly visible throughout the area of interest.
- There should be no foreign objects (wires, road signs, trees, etc.) that can cover the number in the camera field of view.
- The vehicle in the camera field of view should move in one direction, i.e. there should be no turns on the road.
- License plate illumination is required for night and dark operation conditions.

> **Note**
>
> If only some part(s) of the frame comply with the above conditions, you can set the detection zone when configuring LPR so that the troubled areas were excluded from recognition process.

Below are examples of correct and incorrect installation of cameras.

Frame example	Description
	Correct installation example
	Poor installation example: the license plate is closed by a wire, the license plates are too small
	Example of poor installation for night conditions: the license plate is fully flared. The vehicle will be detected, but the license plate will not be recognized, although its presence or absence cannot be confirmed. Additional illumination is required.

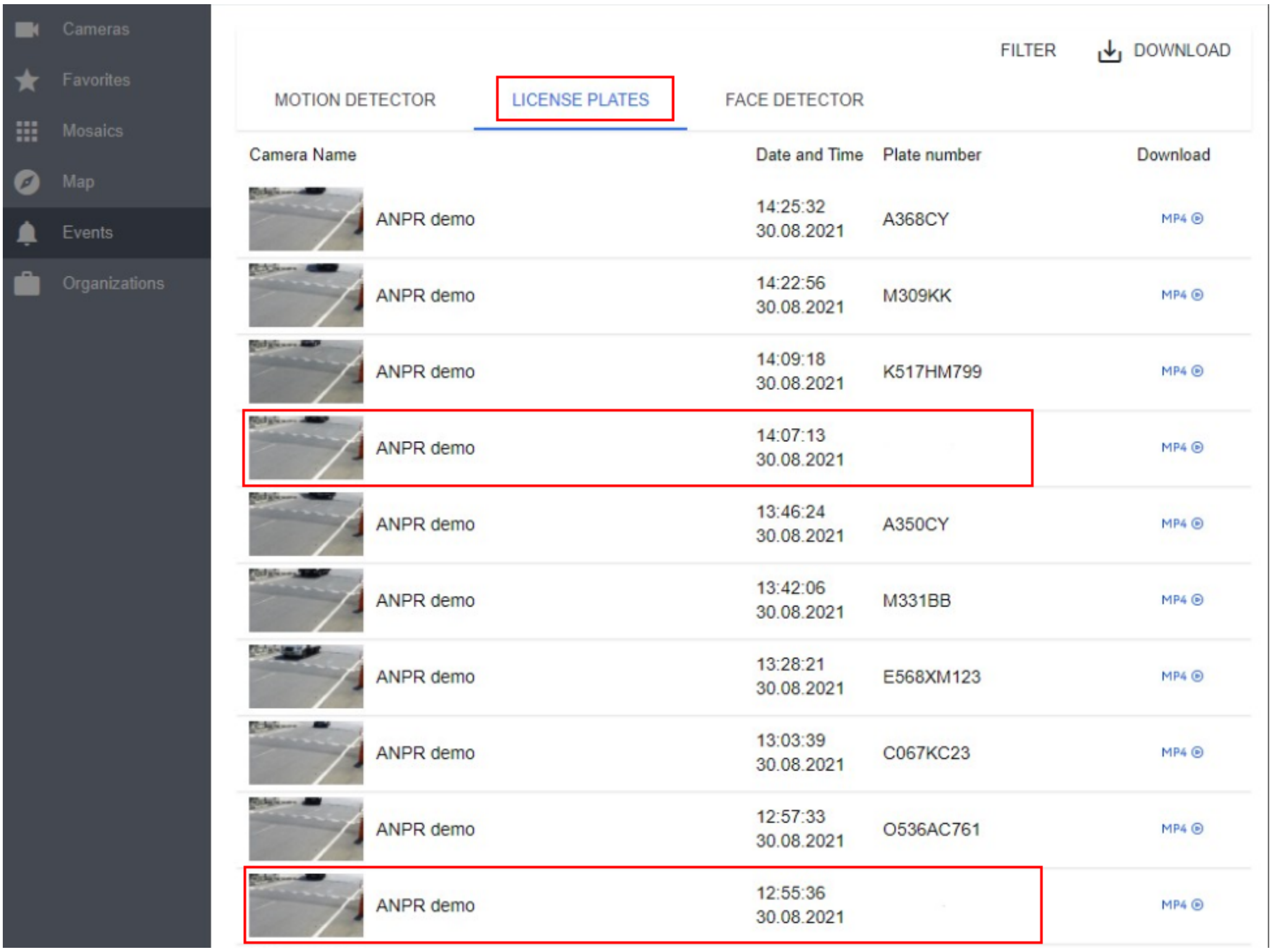
Enabling the detection of vehicles without license plates

To enable detection of vehicles without license plates, just install and configure the LPR module.

> ⚠ **Warning**
>
> If a certain lane is not completely visible in the field of view, make sure to adjust the detection area so as to exclude this lane from recognition. Otherwise, all vehicles in the partially visible lane will be detected as vehicles without license plates because the detector will detect the car but will not detect the license plate.

Browsing and searching for vehicles without license plates

The detected vehicles without license plates are included in the general list on the **Events** -> **License plates** tab without value in the **Plate number** column.



Camera Name	Date and Time	Plate number	Download
ANPR demo	14:25:32 30.08.2021	A368CY	MP4
ANPR demo	14:22:56 30.08.2021	M309KK	MP4
ANPR demo	14:09:18 30.08.2021	K517HM799	MP4
ANPR demo	14:07:13 30.08.2021		MP4
ANPR demo	13:46:24 30.08.2021	A350CY	MP4
ANPR demo	13:42:06 30.08.2021	M331BB	MP4
ANPR demo	13:28:21 30.08.2021	E568XM123	MP4
ANPR demo	13:03:39 30.08.2021	C067KC23	MP4
ANPR demo	12:57:33 30.08.2021	O536AC761	MP4
ANPR demo	12:55:36 30.08.2021		MP4

To search for events of vehicle without a license plate recognition, you can use an API request like **GET http://WATCHER-IP/vsaas/api/v2/analytics/license_plates** with the `event_type=no_license_plate` parameter.

Example:

```
curl -v GET -H 'x-vsaas-api-key: 7c75da8fb314183f1f825271898a3687' \
-d 'limit=12&offset=12&event_type=no_license_plate' \
-H "Content-Type: application/x-www-form-urlencoded;charset=UTF-8" \
http://127.0.0.1/vsaas/api/v2/analytics/license_plates
```

Put your own API key to the x-vsaas-api-key parameter or use another way to authorize the API request.

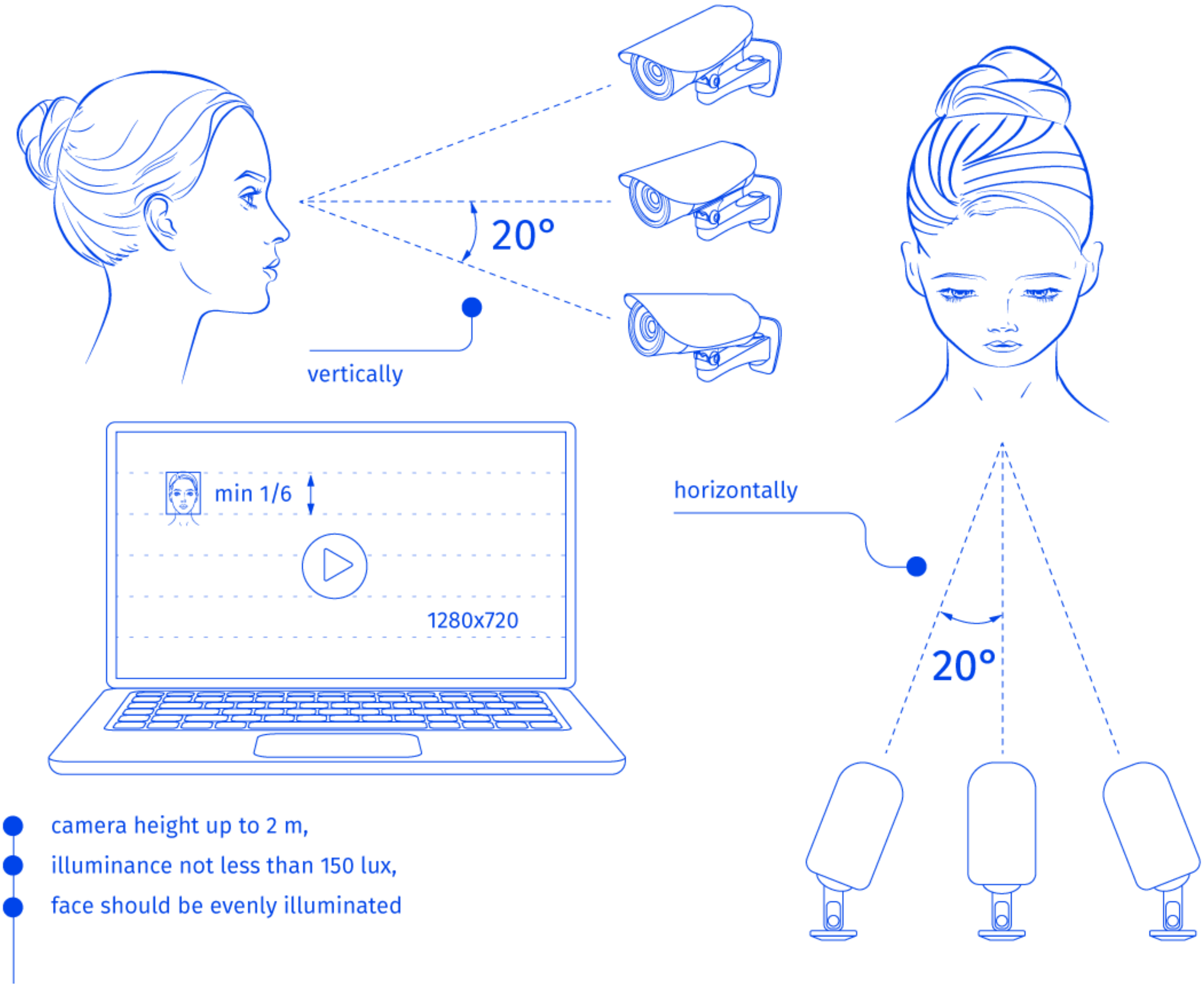
4.4.5 Face Recognition

Flussonic Watcher supports face detection and recognition. *Face detection* is finding a face on a video frame. *Face recognition* is matching detected faces with the *person* (face sample) database to answer the question "Who is it?".

Recommended video image parameters for face recognition

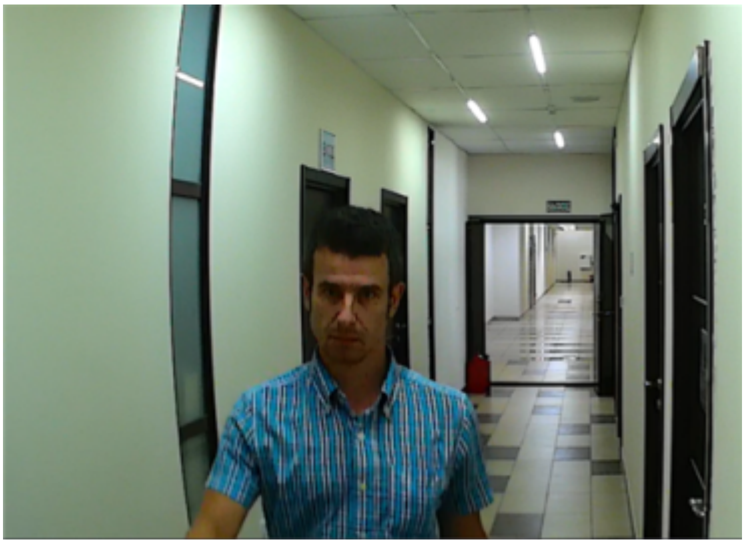
A general requirement for a video image for face recognition is the ability to recognize the face with human eyes. In other words, if you do not perceive the face in the video, then Watcher will not be able to recognize it either.

Stable face recognition is guaranteed with the following video image characteristics:

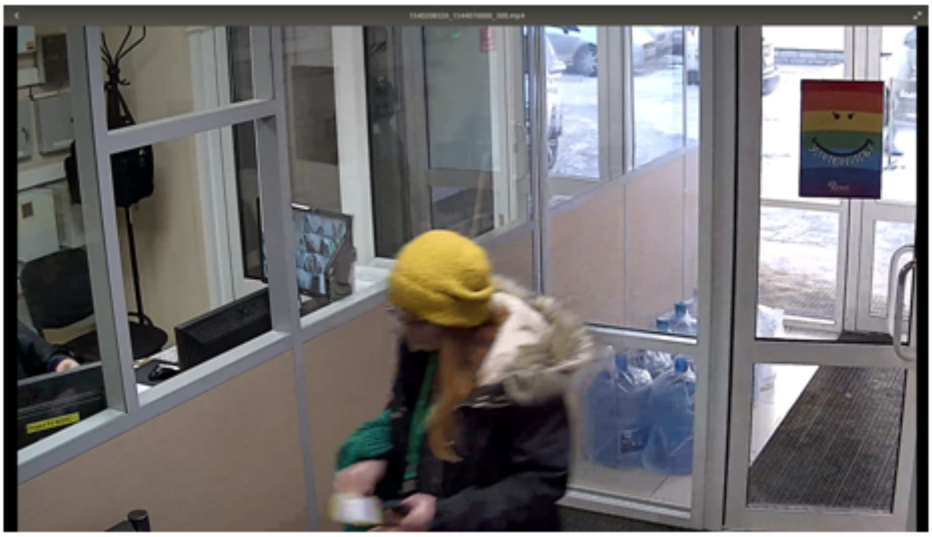

20°
vertically
min 1/6
1280x720
horizontally
20°
camera height up to 2 m,
illuminance not less than 150 lux,
face should be evenly illuminated

1. No more than +/- 20° vertical and horizontal deviation from a right-angle (90°) **view into the camera**, i.e. camera should be installed at a height of no more than 2 m.
2. **Shutter speed** not more than 1/250 to avoid noise in the frame and faces blurring.
3. **Frame resolution** at least 1280x720 (720p). If the frame has larger resolution, it will be reduced to 720p before feeding to analytics module. Also, if camera has several streams, the closest to 720p will be used for analytics purposes.
4. **Bitrate** at least 2 Mbps.
5. The **face height** should be at least 1/6 of the frame height at the resolution of 1280x720. For higher resolutions, the face size should be larger in proportion to the increase in frame size. In other words, resolution should be 500 pixels per 1 m, i.e. the distance between the pupils must be at least 50-60 pixels at 720p.
6. **Lighting** not less than 150 lux with face evenly illuminated.

Example of proper camera position



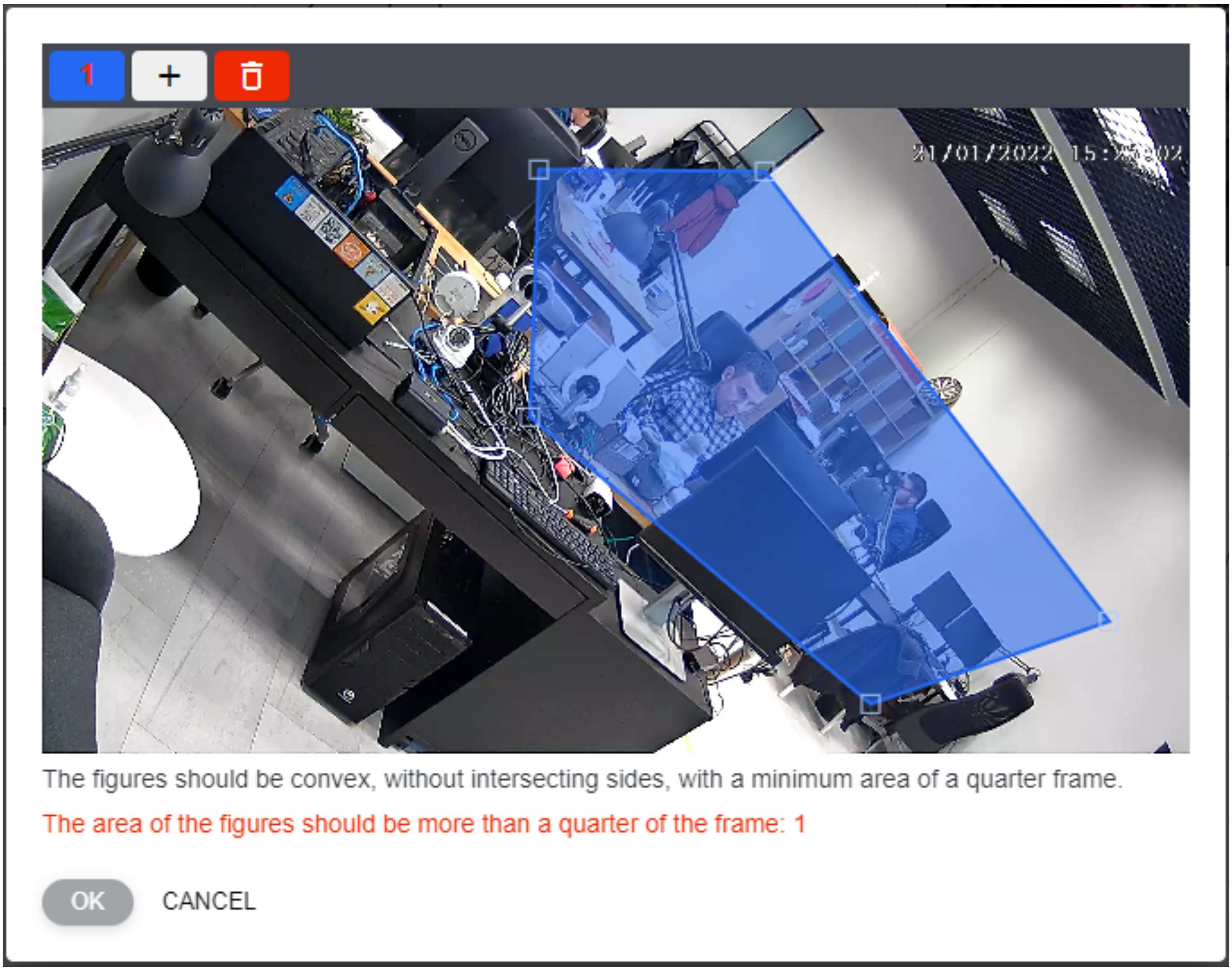
Example of bad camera position



However, if your video image does not meet these guidelines, you can check to see if recognition will work. If the face recognition quality turns out to be unsatisfactory, we may consider the module modification to suit your conditions. Please contact our technical support team by following these instructions. Technical support team member will request the necessary information (for example, access to the stream you use for recognition) and will let you know if the improvements are possible and when we will implement them.

Detection zone

By default, the recognition system searches for faces over all the camera field of view. You can select specific polygonal area(s) to detect faces in by clicking **Set up the detection zone**. This settings may help you to reduce false detections. Areas are set individually for each camera, even if you select a non-configurable preset.



The figures should be convex, without intersecting sides, with a minimum area of a quarter frame.

The area of the figures should be more than a quarter of the frame: 1

Lists of persons

Flussonic Watcher supports face recognition in 1:N identification mode, that is, it allows you to find out which person corresponds to the face detected in the frame. When you enable face recognition on the camera, Flussonic Watcher starts detecting faces, and if there are lists of faces, also recognize the detected faces.

For face recognition to work, you should populate the person database in Watcher by creating persons and lists of persons. A *person* is a sample face saved in the Watcher database with accompanying attributes, such as ID, name, etc. A *list of persons* is a set of persons united on some logical basis, for example, people from the same department or having the same access level to the premises.

You can create one or several person list(s) and add several persons to them. As the persons pass under the camera, events about their passages will appear in the system on the **Events** tab. If a face is recognized, then the person’s name will be indicated in the event, and if the face is detected but not recognized, then you will see an identifier of a newly created unknown person. You can edit such persons later or create new persons.

Each camera can recognize persons from one person list only. Both recognized and unrecognized persons are included in the lists linked to the corresponding cameras. If a face is detected or recognized on a camera that is not linked to any list, then the person is included in the **Persons not on any list** list.

Required permissions

The following user permissions are required to manage persons and person lists:

- Watcher Administrator can view, edit, or delete all persons and person lists in the system; OR
- Organization Administrator (owner) can view, create, edit, or delete only the persons and person lists within the owned Organization (first of them if many).

Integration with third party systems

With API, you can use the face recognition events in other systems (for example, when integrating with access control systems, when you want to allow only employees of a particular office to enter the door).

Use the **External ID** for integration with Access Control System or other third party system. At such integrations, Watcher usually have to transmit message about the person it has recognized. However, person databases in Watcher and external system are not related whatsoever. This is why you should enter the external ID of each person i.e. the ID person has in the external database. This will help the external system to understand which person the message from Watcher is related to and proceed with granting/denying access according to its internal logic or performing other actions.

4.5 Watcher Settings

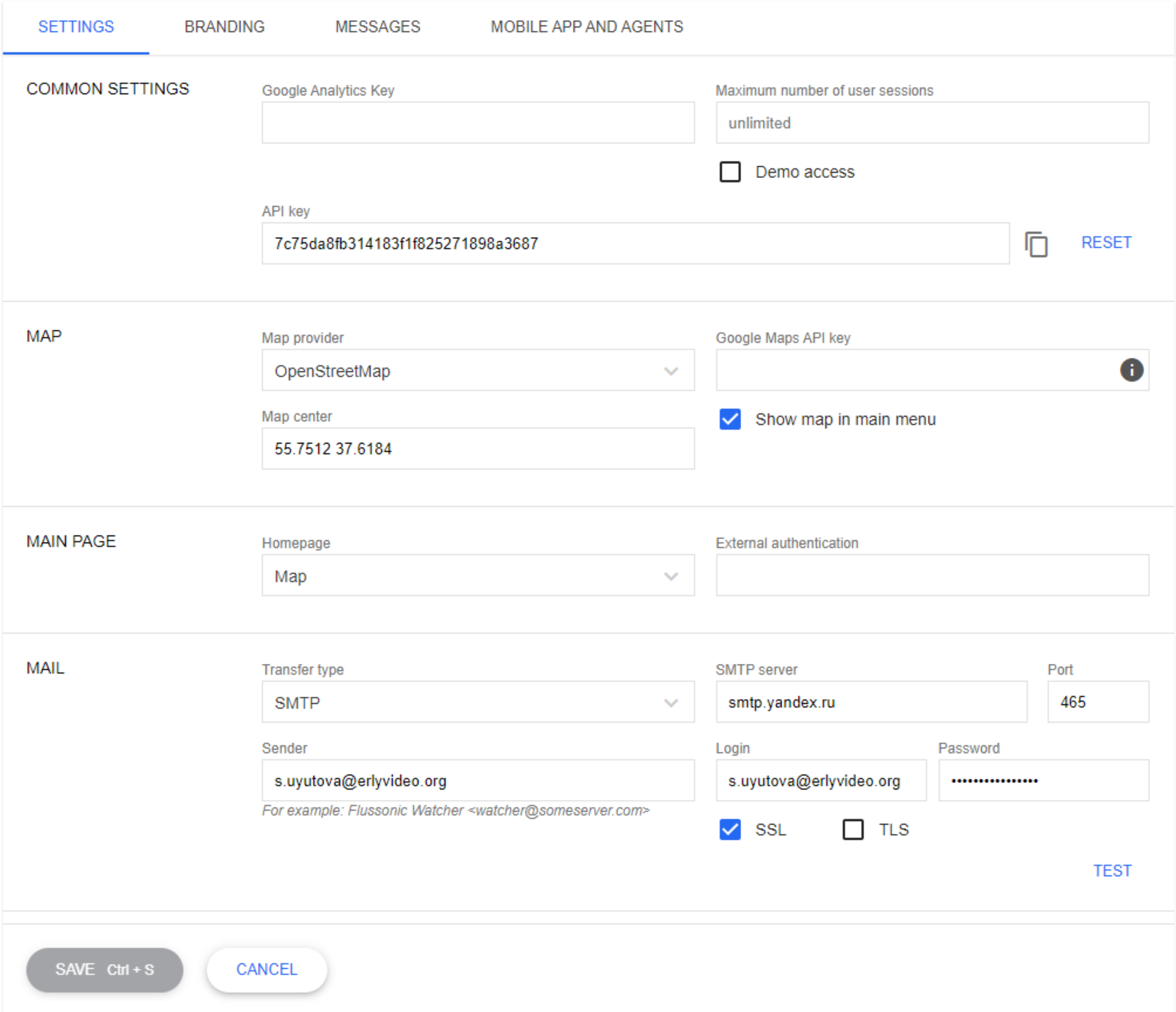
4.5.1 Settings

The **Settings** menu section in the Watcher web interface allows the Watcher administrator to edit settings of Flussonic Watcher.

On this page, you will find the description of parameters on the **Settings** tab. Please refer to the following pages for details on other tabs and other related parameters:

- Web interface appearance settings
- Mobile applications
- Cluster status
- Adding and configuring streamers

Common settings



SETTINGS BRANDING MESSAGES MOBILE APP AND AGENTS

COMMON SETTINGS

Google Analytics Key:

Maximum number of user sessions: unlimited

□ Demo access

API key: 7c75da8fb314183f1f825271898a3687

RESET

MAP

Map provider: OpenStreetMap

Google Maps API key:

Map center: 55.7512 37.6184

☑ Show map in main menu

MAIN PAGE

Homepage: Map

External authentication:

MAIL

Transfer type: SMTP

SMTP server: smtp.yandex.ru

Port: 465

Sender: s.uyutova@erlyvideo.org

For example: Flussonic Watcher <watcher@someserver.com>

Login: s.uyutova@erlyvideo.org

Password: •••••••••••••••

☑ SSL □ TLS

TEST

SAVE Ctrl + S CANCEL

- **Maximum number of user sessions** – the limitation to the number of user (subscriber) sessions. There is similar parameter in user settings which has higher priority.
- **API key** — this key is required for mobile apps and Agents operation. You can also use it to authorize API requests, see API request authorization ways for details.
- **Google Analytics Key** — your Google Analytics key for Watcher. With Google Analytics you can gather statistics on Watcher usage by users.
- **Demo access** — allows access in demonstration mode by the **Demo access** link on the login page. Please refer to Demo access to cameras for details.

Map

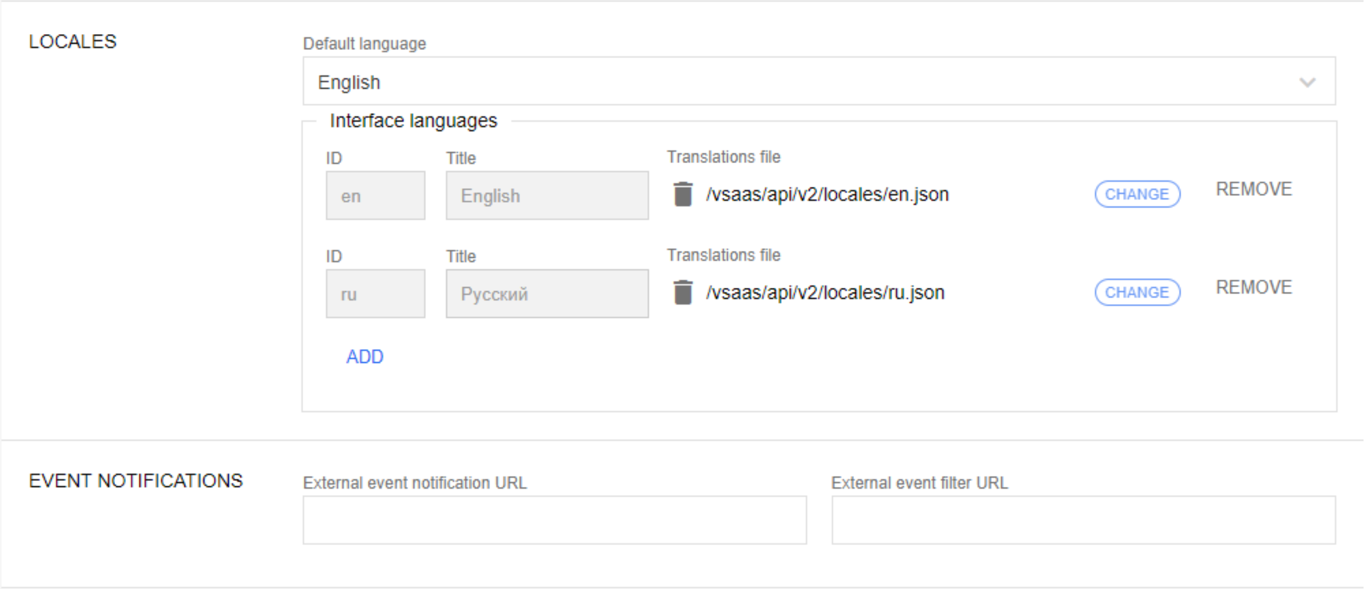
- **Show map in main menu** — show or hide the **Map** section from the main menu. The map is turned on by default.
- **Map center** — geographic coordinates for map centering.
- **Map provider** — select the map provider: Google, Openstreetmaps or OpenStreetMaps Offline.
- **Google Maps API key** — the token of the geo cover in use (Google API key), allows automatically translate an address into coordinates in order to show the camera on the map.

Main page

- **Homepage** — specifies what users will see upon login: map or dashboard.
- **External authentication** – specify the HTTP address or address of RADIUS server that you use to authenticate users.

Mail

See Email customization.



LOCALES

Default language: English

Interface languages

ID	Title	Translations file
en	English	/vsaas/api/v2/locales/en.json
ru	Русский	/vsaas/api/v2/locales/ru.json

EVENT NOTIFICATIONS

External event notification URL

External event filter URL

Locales

- **Default language** – select the default language of the web UI. If not selected, Watcher will use your web browser language.
- **Interface languages** – controls to change, add or delete you own localization files.

Event notification settings

You can select which events received from cameras Watcher will to process. Also, you can instruct Watcher to receive events sent from external systems. License plate and face recognition events are supported. (Usage examples will be added here soon.)

- **External event notification URL** — Watcher will send HTTP events to external sources specified here.
- **External event filter URL** — the URL of your custom event handler. Watcher automatically sends the events received from cameras to this URL at the moment when an event takes place. Your script then receives an event from Watcher and returns (or doesn't return) the event ID. If the ID is returned, the event is considered confirmed and Watcher registers it in its database. Also, an email notification and push notification to the mobile application are sent. The archive interval around the event ([event_utc-10, event.utc+30]) is protected from deletion.

  If the script returns no identifier, the event is considered unconfirmed and is not registered in Watcher.

  If this field is not filled, then all events are registered in the Watcher.

  To sum up, this script is for your custom filtering of events. You can save only those events that you are interested in.

  Watcher uses the JSON format to send info about events to your handler:

  ```
  {
       "event":"video_activity",
       "camera_id":"test1",
       "algorithm":"plate_detector",
       "activity_type":"enter",
       "number":"ABCDEHKMOTX",
       "area_id": "0",
       "start_at": 1554883886,
       "end_at": 1554883886
  }
  ```

  Example of the Simple Event collector.
- **Disable push notifications** — notifications about video analytics events (such as motion events) will not be sent to the Watcher mobile app. This option is independent from your custom backend scripts.

  Example of the Simple Event collector.

Adding and configuring streamers in the UI

Before you add streamers to Watcher cluster, you need to prepare them as described here. If you use Watcher single, just configure the local streamer as needed.

4.5.2 Adding and configuring streamers

Streamer list

The **Streamers** menu section in the Watcher web UI allows the Watcher administrator to monitor streamer statuses as well as configure and add streamers. By default, the local streamer is added to Watcher.

Adding and configuring streamers

Before you add streamers to Watcher Cluster, you need to prepare them as described here. If you use Watcher Single, just configure the local streamer as needed by clicking its name to open the form described below.

> **Note**
>
> If you are adding the streamer for analytics, refer to video analytics installation guide.

To add streamer servers to Watcher:

1. Click the **+** button. In the streamer creation form, enter the hostname (actual URL or any name), cluster key, and select the **Streamer** role.

   When you click **Save**, the streamer settings page appears.

2. Specify the streamer settings:

   - **Role** of the streamer in your cluster. Use Streamer for cameras ingest, Inference or Identification for video analytics.
   - **Issue by LetsEncrypt** is to issue an SSL certificate so that the streamer could be accessed via HTTPS.

     > **Note**
     >
     > Issue the certificate for the localhost streamer to use HTTPS with the managing server.

   - **Default** – when you have added a number of streamers, you can select the default one. All new cameras will be added to the default streamer(s).
   - **API URL** – this URL is used for internal communication between the Watcher server and the streamer. It shall conform to the specifications of the FQDN.
   - **Public payload URL** – this URL is for connecting with streams at the streamer when you use the embed link. Note that the streamer's public URL must start with `https://` if you plan to use the embed links over HTTPS because the public URL is inlined in the player page and you cannot call HTTP URLs from HTTPS page.

     The public URL shall conform to the specifications of the FQDN.
   - **Cluster key** – the cluster key used in the cluster (the cluster_key option in the configuration file). If the streamer’s and Watcher's cluster keys are identical, there is no need to fill this field.

3. Click **Save**.

The streamer appears in the list. The green mark next to the hostname indicates that it is connected.

> **Warning**
>
> Identical date and time must be set on all servers in the cluster.

DVR settings

To record an archive, configure it on the **DVR** tab in the streamer profile. Specifying **Path** where the archive should be stored is enough to enable recording on the streamer.

You can also configure a software RAID array that allows for distribution of archive fragments across disks ensuring uniform degradation of the records in case of disk failure. Thus, you can avoid losing all the archive of any particular camera.

Please refer to the schema for the parameter descriptions.

4.5.3 Failover

A failover cluster is a group of servers that work together to maintain the overall service stability and exclude any possible downtime of any part of the system. If one of the servers fails, another cluster server takes over its workload. This process is called *failover*.

In Watcher cluster, in case of a streamer server failure, the camera traffic to this server will automatically redistribute between other cluster servers, also called standby servers. The video archive becomes unavailable on the streamer server that had a failure. The new video archive will be recorded on the standby server.

⚠ **Warning**

When the connection to the primary server that had an issue is renewed, the traffic isn't automatically redirected back. Traffic can be returned back to the main streamer if necessary for proper resource distribution.

Once the server is restored, provided that the storage on the main server was not damaged during the file, access to the main video archive will be restored.

The failover function is enabled in Watcher by default. No additional settings are required.

4.5.4 Reset password

A user can use the **RESTORE PASSWORD** option on the login page of Watcher UI. A password recovery link will be automatically sent via email. Read how to configure an SMTP server.

An administrator can change the user password with the `watcher` utility.

Example:

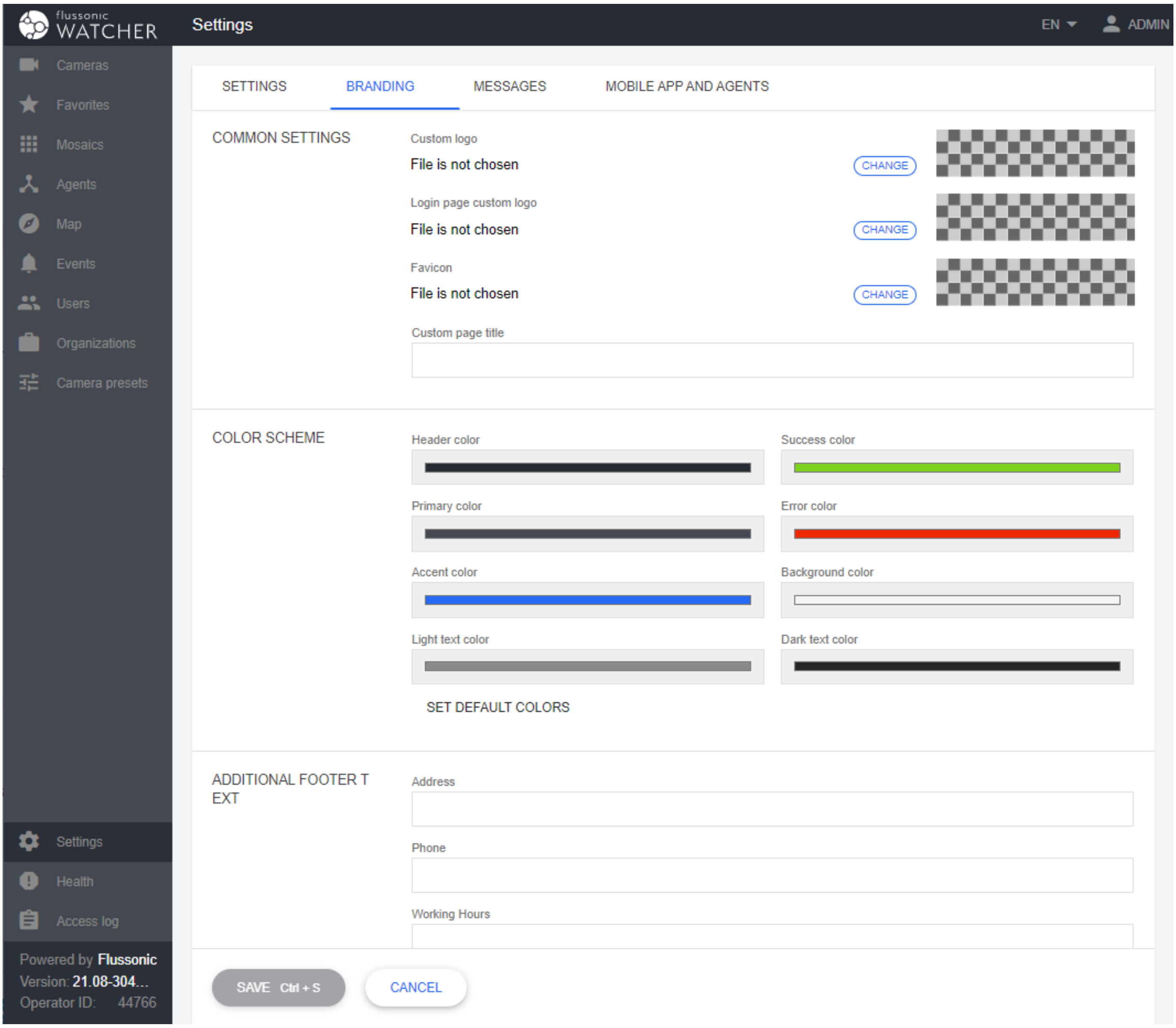
```
/opt/flussonic/contrib/watcher reset_pass support@erlyvideo.org new_password
```

You will see this message if everything went OK:

```
Changing password for support@erlyvideo.org
```

4.5.5 Interface Customization

Watcher in cluster mode provides tools for interface customization (branding).



COMMON SETTINGS

Custom logo
File is not chosen

Login page custom logo
File is not chosen

Favicon
File is not chosen

Custom page title

COLOR SCHEME

Header color

Success color

Primary color

Error color

Accent color

Background color

Light text color

Dark text color

ADDITIONAL FOOTER T
EXT

Address

Phone

Working Hours

Powered by Flussonic
Version: 21.08-304...
Operator ID: 44766

To set your custom branded interface for Watcher:

Go to **Settings** > **Branding** and specify:

Common settings

- **Custom Logo.** Select a graphic file with a logo image to be displayed in the upper right corner. If the file is too large in width and height, the system will reduce it to the required size.
- **Login page custom logo**. Select a graphic file with a logo image to be displayed on the page where you enter the login and password. If the file is too large in width and height, the system will reduce it to the required size.
- **Favicon**. The icon that appears in the browser on the tab where Watcher is open. The favicon must be a square PNG image of 64x64 pixels.
- **Custom page title**. The title that appears in the browser on the tab where Watcher is open.

Color scheme

You can select colors for basic UI elements, and Watcher will define all other colors automatically based on the specified main colors.

Additional footer text

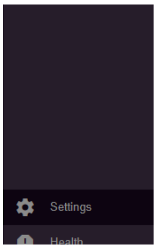
- **Address**. Your company postal address.
- **Phone**. Your company phone number.
- **Work hours**. Your company working hours.

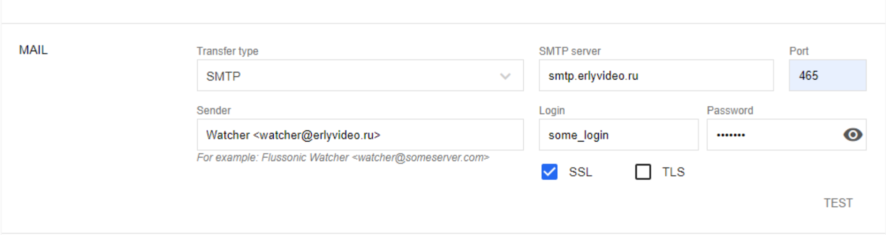
You can also customize the password recovery email template. Read more in Email customization.

4.5.6 Email customization

Setting the SMTP server

Configure your mailbox settings so that password recovery messages reach your users. Add outgoing mail server parameters in UI Watcher settings





MAIL

Transfer type: SMTP

SMTP server: smtp.erlyvideo.ru

Port: 465

Sender: Watcher <watcher@erlyvideo.ru>

For example: Flussonic Watcher <watcher@someserver.com>

Login: some_login

Password: •••••••

☑ SSL ☐ TLS

Warning

Both login and password must NOT include any of the following characters: @, ;, #, [, \, /

Changing the template of password recovery emails

To change the template for password recovery emails, follow these steps:

- Go to `/opt/flussonic/lib/vsaas/watcher/templates` .
- Use `password_reset_request.email` and `password_changed.email` text files as an examples.
- Add `custom_ file prefix` to your custom template versions: `custom_password_reset_request.email` and `custom_password_changed.email` .
- If you want to use custom hypertext templates, add the `.html` extension to these files as follows: `custom_password_reset_request.email.html` and `custom_password_changed.email.html` . Save these files in the same directory.

The template consists of two parts: the header and the body. You can also add a subject to the header.

In addition, in the message body you can use variables

- {{data.base_url}}
- {{data.token}}

Text template example:

```
custom_password_reset_request.email:
---
subject: "ABC surveillance password reset"
---

Thank you for using the forgot password option. Follow this link to reset your password to the ABC surveillance system:
{{data.base_url}}/vsaas/forgot-password/{{data.token}}
```

HTML template example:

```
custom_password_reset_request.email.html:
---
subject: "ABC surveillance password reset"
---
<html>
<body>
<p>Thank you for using the forgot password option.to the ABC surveillance system:<br>
<a href="{{data.base_url}}/vsaas/forgot-password/{{data.token}}">Reset your password</a>
</p>
</body>
</html>
```

4.5.7 Motion Detection Events Processing

The server Flussonic Watcher can receive events over the **SMTP** protocol. Cameras send motion detection events over this protocol, and Watcher adds corresponding marks in the archive recordings in the places when motion was detected.

> **Note**
>
> Watcher can also receive motion events via ONVIF. To enable this detection mode, go to the cameras ONVIF settings in Watcher.

How motion recording works

Flussonic Watcher continuously keeps recording of video received from a camera, with the specified archive depth. When an event arrives, Flussonic Watcher saves the time interval in the database to be able to show the event in the archive player. The record with detected motion is **protected from deletion**.

The duration of a protected recording is determined by two timestamps, the first of which is calculated as the current time minus 10 seconds, and the second timestamp is the current time plus 30 seconds.

You need to set the depth of the archive, for example, 6 hours, and then enable the reception of events. As a result, you will have a recoding of 6 hours of continuous archive and additionally motion events, which will be stored as long as there is free space on the disk. Recording of new events will delete the old ones.

By calculating the necessary disk space based on the bitrate of the cameras and the frequency of motion events, you can save up to 50-90% of the disk space compared to the normal recording without events.

On this page

- Configuring motion detection events for cameras with Flussonic Agent
    - Configuring the camera
- Viewing events in the web interface of Flussonic Watcher

Configuring motion detection events for cameras with Flussonic Agent

If you have Flussonic Agent installed on a camera to automatically connect the camera to the server, if it is behind NAT, then you do not need to configure anything on the server. It is enough to configure the camera to send events.

To configure the camera, specify the SMTP server address, and the names of sender and recepient.

Go to the camera's interface to the message sending section and specify SMTP server settings:

```
SMTP Server: 127.0.0.1
SMTP Port: 5025
```

Fill in other fields. Sender name is the name of the camera.

> **Note**
>
> You can specify only a part of the camera name if the camera has Flussonic Agent installed. The full name will be provided automatically.

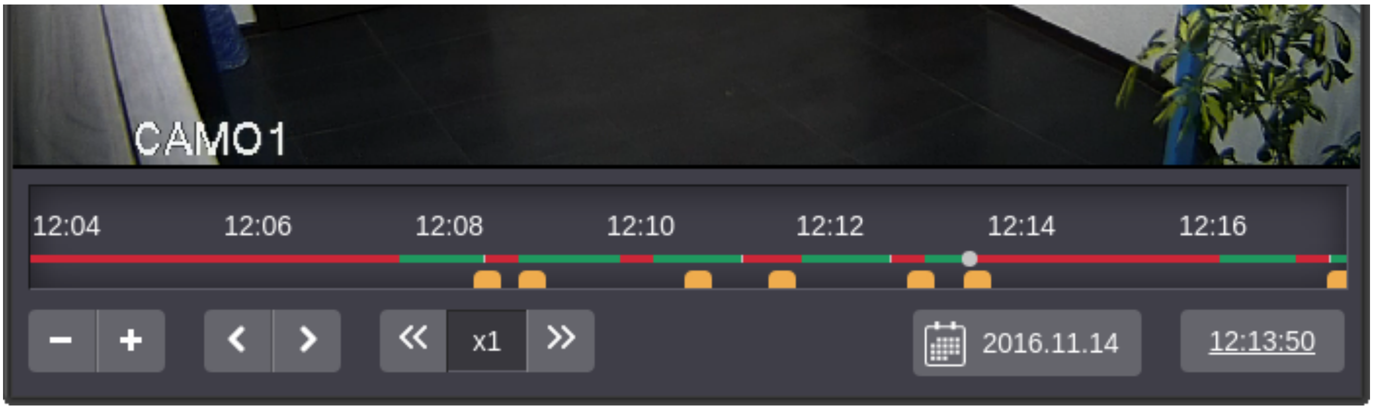
Viewing events in the web interface of Flussonic Watcher

After you have configured a camera to send motion events, you can view the events in the DVR archive.

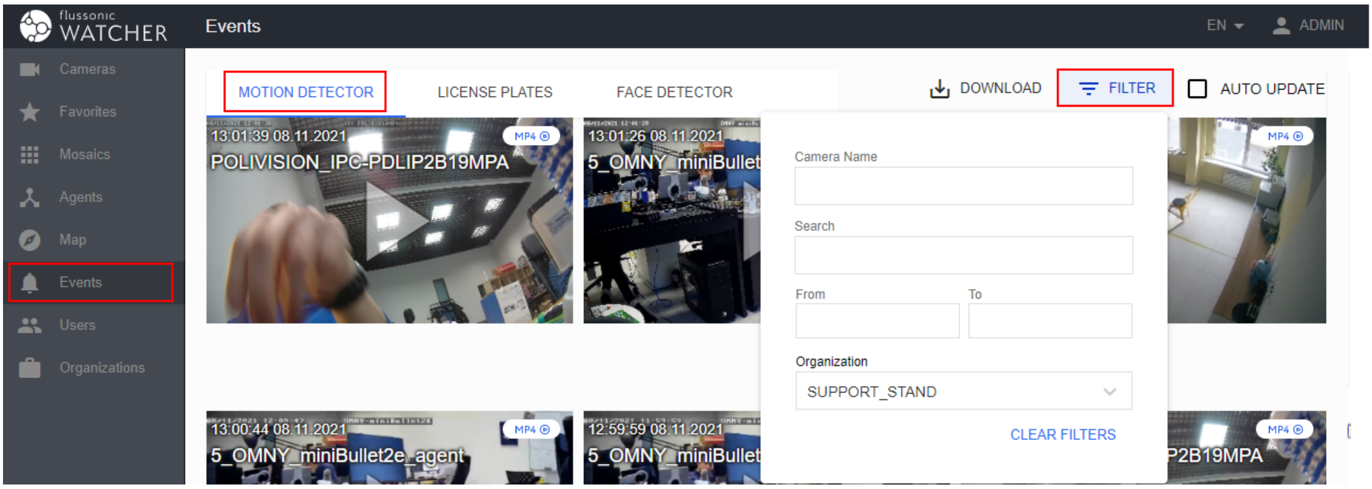
To view the DVR archive of the camera:

1. Go to **Cameras** in the Watcher main menu.
2. On the top of the page, select the List mode for viewing the list of cameras.
3. Find the camera (for example, by using the search form on the right).
4. Open the actions menu for this camera by clicking the icon in the most right column.
5. Choose **View**. The player opens and you can jump to the interval with recorded video. If DVR is disabled for a camera, the player shows only live video.

If there were motion events, you will see the marks on the timeline that show points in time when the camera recorded movement:



You can also go to the **Motion detector** tab on the **Events** page to view the events from all cameras. Use **Filter** to search for an event e.g. from specific cameras or in specific time. Click the event tile to view the event recording.



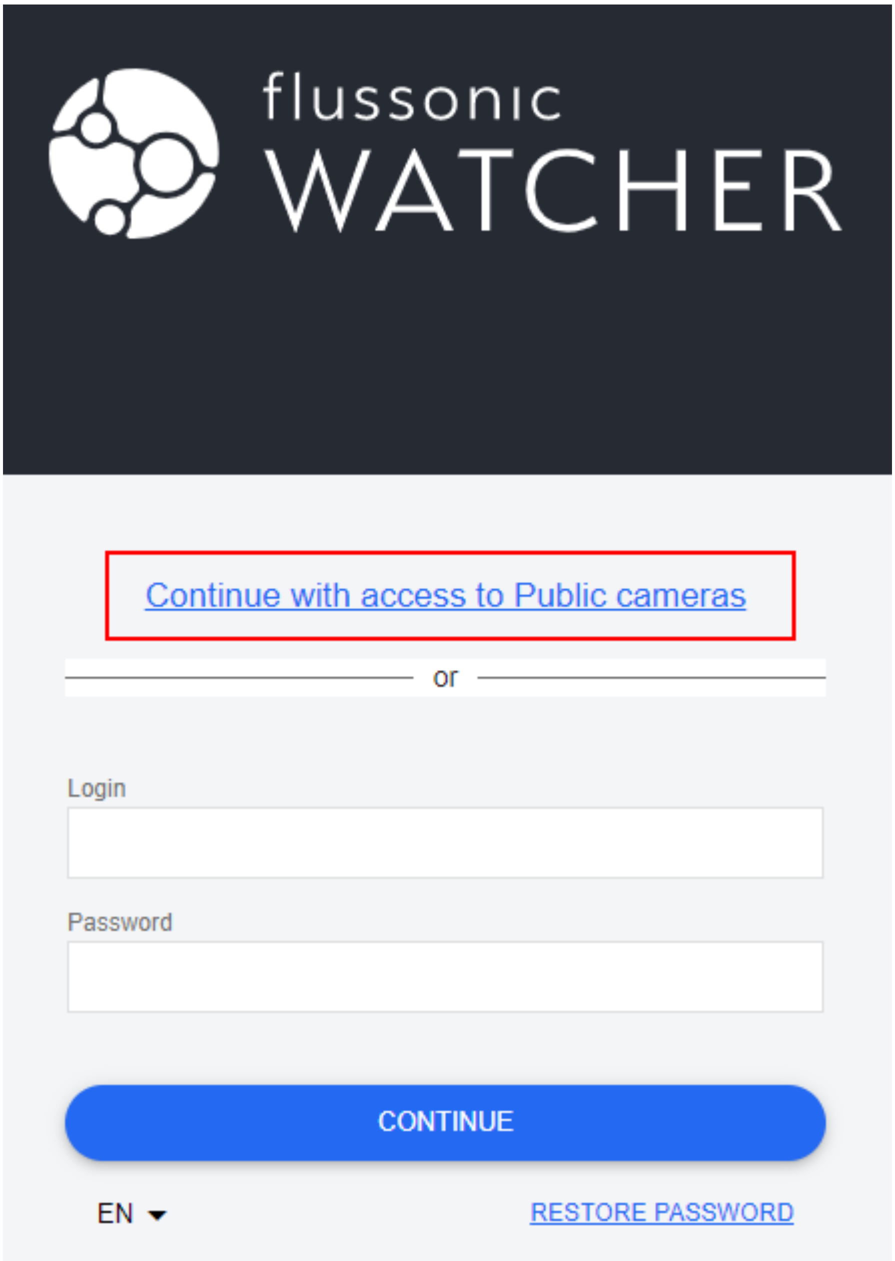
See also Events.

4.5.8 Public access to cameras

You can provide unregistered users with public access to Watcher web UI and selected cameras in it. This feature can be useful, for example, if you have some publicly available cameras on crossroads and private cameras in housing estates. In this case you can allow unregistered users to view public cameras for free and then convert those users to subscribers when they wish to get access to cameras in their housing estate. Thus, public cameras will help attract new clients to your service.

> **Note**
>
> It is also possible to share individual camera(s) by direct link as described in Viewing video from cameras on external websites

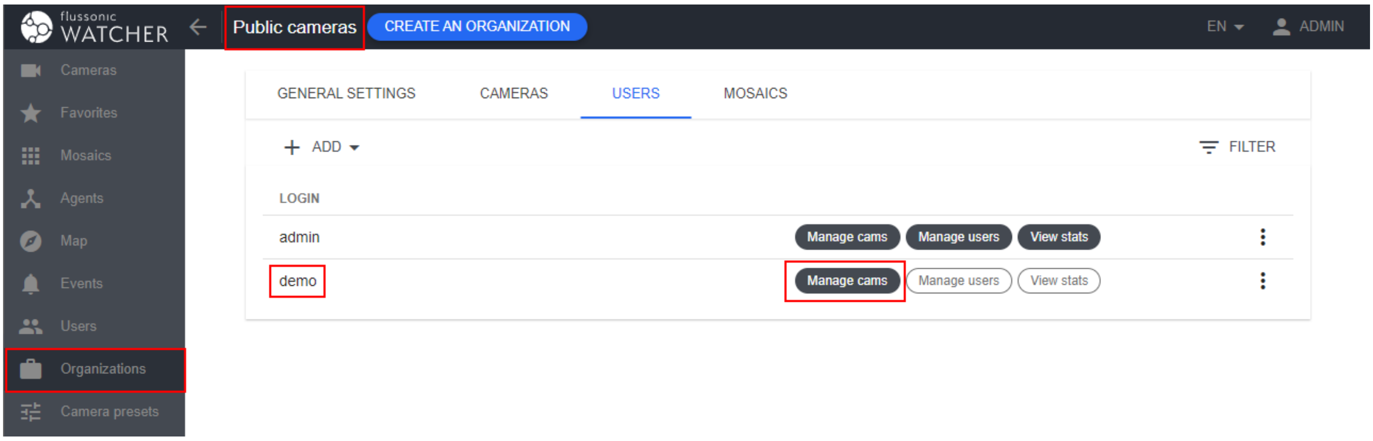
Just check the **Demo access** box in the Watcher settings. Right after you save this setting, the **Continue with access to Public cameras** link becomes available on the Watcher login page.



When anyone clicks this link, a session for the Demo user is initiated. The Demo user is a designated user that is created in Watcher for demo access when you check the box.

However, the Demo user has no permissions by default and is not added to any Organization, so they will see empty camera list at login. You should go to **Users** (or **Organizations**) page to assign the Demo user with some permissions.

We recommend that you create a special Organization with public cameras added to it and just add the Demo user to this Organization with permissions to manage cameras only. Please refer to User and permission management for details on permissions that you can assign. Please note that the Demo user has read-only permissions by default, and you cannot change Demo user's password: even if you set one, it will not be applied.



LOGIN	
admin	Manage cams Manage users View stats
demo	Manage cams Manage users View stats

4.6 Using Flussonic Agent

4.6.1 Using Flussonic Agent

Flussonic Agent (or simply Agent) is a small-size software that allows cameras from a local area network (LAN) to connect with an external Watcher server via an encrypted channel. The Agent can be added to the IP camera firmware by manufacturer.

For the Agent on camera to connect with the server, it is enough to connect the camera to your LAN with access to the Internet: IRIS firmware allows this with a QR code, in other cases please refer to the manufacturer's guidelines.

The benefits of using Agent

Agent solves the problem of communication between Watcher and devices from behind NAT. It helps if the camera does not have a dedicated IP address or if you do not want to perform port forwarding on network equipment so that the camera in the local network could be seen by a remote Watcher server via the Internet. With Agent, a camera itself initiates the connection with Watcher and automatically registers there (while without Agent, usually it is the server that initiates the connection with cameras).

In addition, Agent helps if the communication channel between the camera and the server is unstable. Many cameras do not support video buffering. Agent installed on cameras creates the buffer to resend packets that for some reason did not reach the server.

The Flussonic Agent concept and advantages section provides details about how Agent works and how it is better than other ways of delivering video from cameras to Flussonic Watcher.

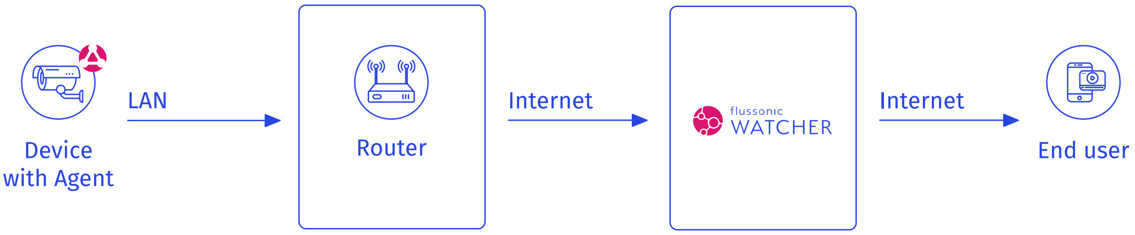
In this section:

- Devices supported by Agent
- Flussonic Agent concept and advantages
- Agent in the Watcher UI
- Monitoring your Agents

Devices supported by Agent

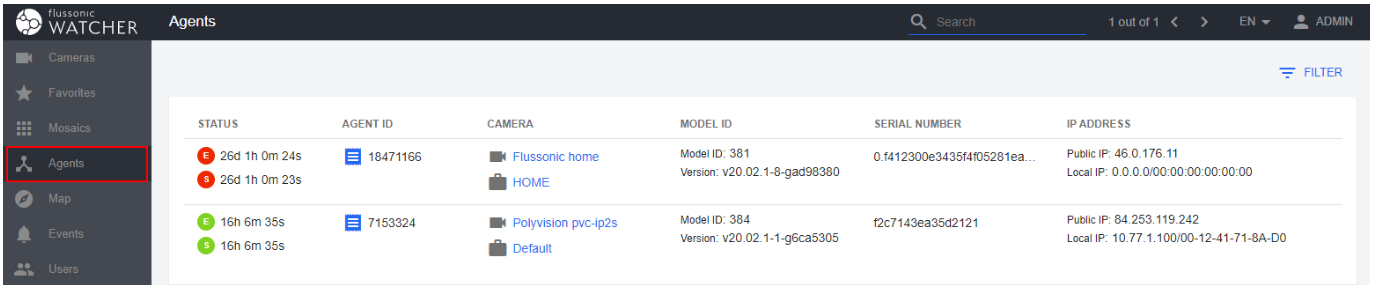
We offer a line of cameras with the Agent already installed. Please check out our branded cameras here or contact our technical support team at support@flussonic.com to get extended list of camera models with Agent. Our team is always ready to help you get the Agent that suits your case.

The camera is connected via Agent as follows:



4.6.2 Flussonic Agent in the Watcher UI

The list of activated Agents registered in your Watcher is shown on the **Agents** page.



STATUS	AGENT ID	CAMERA	MODEL ID	SERIAL NUMBER	IP ADDRESS
E 26d 1h 0m 24s S 26d 1h 0m 23s	18471166	Flussonic home HOME	Model ID: 381 Version: v20.02.1-8-gad98380	0.f412300e3435f4f05281ea...	Public IP: 46.0.176.11 Local IP: 0.0.0.0/00:00:00:00:00:00
E 16h 6m 35s S 16h 6m 35s	7153324	Polyvision pvc-ip2s Default	Model ID: 384 Version: v20.02.1-1-g6ca5305	f2c7143ea35d2121	Public IP: 84.253.119.242 Local IP: 10.77.1.100/00-12-41-71-8A-D0

The Agent can be supplied either already bound or not bound to a specific Operator ID that uniquely identifies the Watcher server URL. Please refer to Mobile Applications page for details on how to find or get the Operator ID.

If the bound Agent is installed on the camera, then it will connect to the Watcher server itself when the camera is connected to the Internet, and you just have to go to the Watcher web UI and check that the Agent has appeared in the list. If the Agent is not bound, then it will appear in the list after you add camera via Watcher mobile app.

The Agent list shows the following data:

- Agent's status — Watcher uses color to show the Agent status (red for not working Agents, green for working Agents). The period during which the Agent was in this status is also shown.
- Agent ID — a unique identifier of this Agent.
- Camera — the name of the camera where the Agent is installed and the name of the Organization to which this camera is added.
- Model ID — information about the camera model and firmware version.
- Serial Number — Agent serial number.
- IP address — local and public IP addresses as well as MAC address.

There is also a **Filter** on this page to search for Agents by Organization to which the corresponding camera belongs.

4.6.3 Flussonic Agent Status and Log

This section describes how to obtain Agent status and Agent log messages.

Agent status

To make sure that Agent has successfully connected to Watcher, you can use the browser (as an alternative to looking at the list of Agents in the Watcher UI).

Open the following address in the browser:

```
http://[AgentIP]:5680/agent-status
```

Replace [AgentIP] with the local IP address of the device where Agent is installed. The current status of Agent will be shown.

> ⚠ **Warning**
>
> Open this address only in the same local network where Agent is installed.

Agent log

For diagnostic purposes, we migth sometimes ask you to send us Agent log messages.

To download Agent logs, open the following address in the browser:

```
http://[AgentIP]:5680/agent-status?k=1
```

Replace [AgentIP] with the local IP address of the device where Agent is installed.

> ⚠ **Warning**
>
> Open this address only in the same local network where Agent is installed.

Save the log messages and attach it to the ticket in which you correspond with our support team.

4.7 Mobile Apps

4.7.1 Mobile Applications

Watcher offers two sorts of applications for real time access to a video surveillance system:

- Traditional mobile applications for Android and iOS devices
- Progressive Web Application (PWA)

Traditional mobile apps

Watcher mobile apps provide:

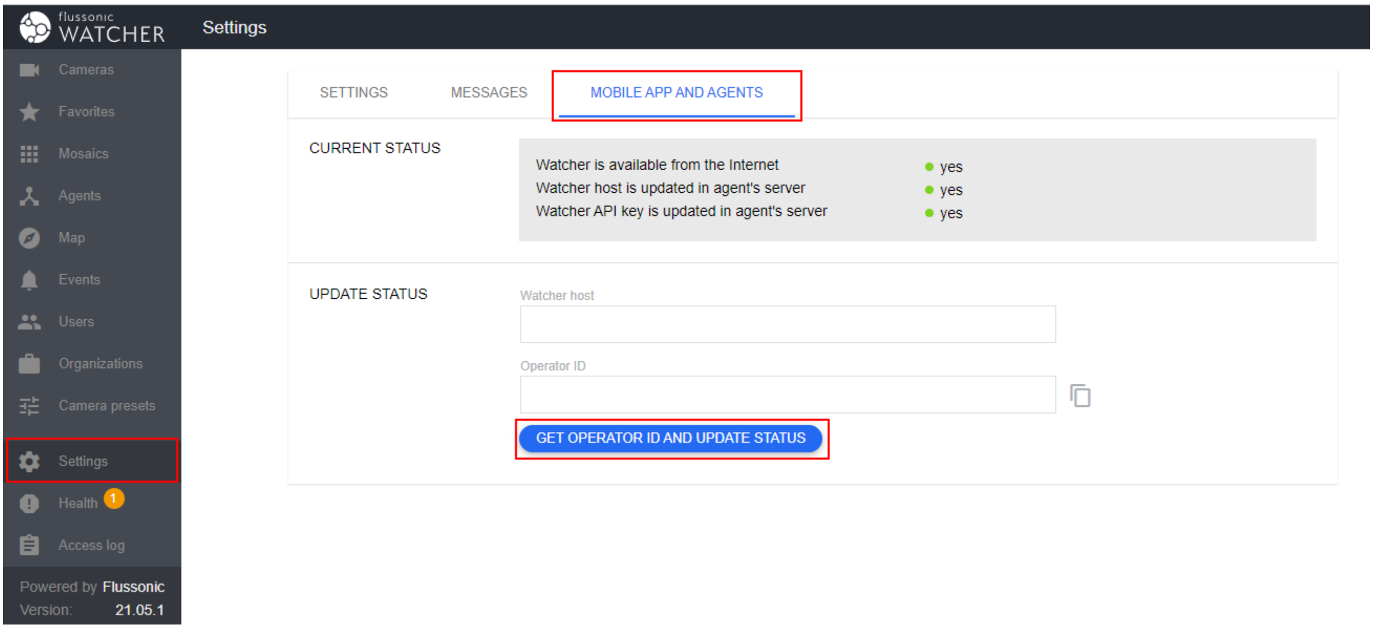
- Provisioning of cameras with Agent to Watcher
- Watching live video from IP cameras with the ultra low latency
- Viewing the archive with no limits on its depth
- The access control based on a fingerprint or a PIN code
- TLS encryption of video streams
- Push-notifications about events
- Downloading video screenshots
- Viewing cameras on the map
- Using mosaics
- PTZ control

GETTING YOUR OPERATOR ID

Mobile applications need to know the address of your Watcher system to get video from it. This is why you should get **Operator ID** before using Watcher mobile apps. Operator ID is a number that encodes the URL of your Watcher system.

To get the Operator ID:

1. In the web browser, go to your Watcher's public (external) URL or IP address accessible from the Internet. Do not use the server's local IP address because the Operator ID will not be issued and the Watcher mobile app will not connect in such case.
2. Log in to Watcher as an administrator
3. In the main menu (on the left of the page), go to the **Settings** section
4. Go to the **MOBILE APP AND AGENTS** tab
5. Click the **GET OPERATOR ID AND UPDATE STATUS** button
6. As a result, your `Operator ID` will appear in the **Operator ID** field.



CURRENT STATUS

Watcher is available from the Internet	● yes
Watcher host is updated in agent's server	● yes
Watcher API key is updated in agent's server	● yes

UPDATE STATUS

Watcher host

Operator ID

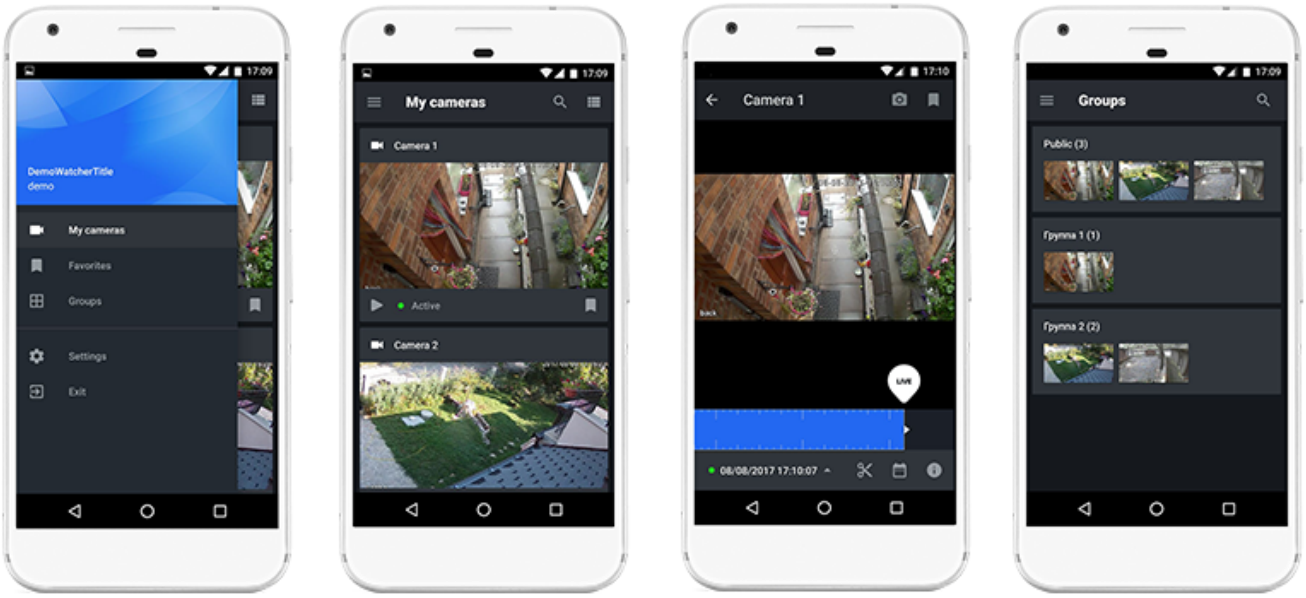
Please note that all **CURRENT STATUS** indicators should be green, otherwise you won't be able to connect your mobile app to the Watcher. One of the reasons for the indicators to turn red is that you have used the local Watcher's address in the web browser, so check that the public URL is shown in the **Watcher host** field. If the URL is correct but one or more indicators are red, try to click **GET OPERATOR ID AND UPDATE STATUS** again. If this does not help, please contact Flussonic's technical support.

> ⚠ **Warning**
>
> Make sure that your server is visible from the Internet and that it has a real domain name. If your server is using NAT or a Firewall, contact our support team and we will help you with all necessary configuring under the terms of extended support.

DOWNLOADING WATCHER MOBILE APP

To start using the application, download it from the **Apple Store** or **Google Play**.

When the app is installed on your device, you can authenticate using your Watcher login, password and the **Operator ID** that you received as described above.



Progressive Web Application (PWA)

Main advantages of using Watcher Client UI as a PWA:

- The same functions are available as in the Watcher Client UI web interface.
- Customized app with your own icon without an app store: just share an URL or QR code with your customers.
- PWA cannot be removed from an app store.
- Automatically updated when you update the server.
- Operator ID is not required for logging in, just login and password.
- PWA can be used as a desktop application on a desktop computer.

INSTALLING WATCHER CLIENT UI AS A PWA

To install Watcher Client UI as a PWA:

1. Open Watcher Client UI in your mobile browser.
2. Select **Add to Home Screen** in the browser menu.
3. Sign in.

4.8 VSAAS.IO video surveillance service

4.8.1 The Billing of the Cloud Service VSAAS.IO

This guide describes how to work with the billing system for the cloud video surveillance service VSAAS.IO.

To get started with the billing system, follow these steps:

1. Signing up and getting started with the billing service
2. Managing billing plans
3. Managing billing users
4. Managing organizations

4.8.2 Signing up and getting started with the billing service

Signing up to VSAAS.IO

The video surveillance service VSAAS.IO allows you to rent all the necessary infrastructure for launching a subscription video surveillance service. To start using the cloud infrastructure, send a request to support@flussonic.com. The request must contain:

- An email of the system administrator
- The domain name to which your service will be connected. For example, yourdomain.com.

Then you should wait for the invitation letter to come to the email address specified in the request.

To activate your VSAAS.IO account, follow the link provided in the email and set your password.

Logging in to VSAAS.IO

With this billing service you can create subscribers and keep record of their payments, create and manage pricing plans. The following employees can have access to the service:

- Service administrators
- Sales managers of the service
- Service accounting Department
- Employees of the client Department
- Subscribers responsible for settlements
- Subscriber's accounting Department

After you have completed the registration process, you need to configure billing.

The billing VSAAS.IO is available at https://my.vsaas.io/.

To log in to billing, use your email address and the password you set when you activated your account.

4.8.3 Managing Billing plans

The first thing to perform is to create billing plans that will be available to your subscribers. Until you create billing plans, your users will not be able to use the camera DVR archive feature, and cameras will only be connected with the ability to view video in real time.

Each billing plan is assigned a group of camera settings in Flussonic Watcher. You cannot change any setting manually via Flussonic Watcher, since it is under Billing control. However, if you change the billing plan for the selected camera, the camera settings will be changed to the required ones.

To open the billing plan management section, go to the **billing plans** in the menu. You will see a list of all billing plans added to the system. If you have added a large number of billing plans, you can use the **Search** box for quick search.

ADDING A BILLING PLAN

1. Create a new billing plan by clicking on the **Create new billing plan** button.
2. Add a new billing plan by filling out the form:

   General information

   - Billing plans name — the name of the billing plan.
   - Note — a note or brief information about the billing plan.
   - Amount — the cost of the monthly subscription fee for this billing plan.
   - Currency — the billing plan's currency.
   - Apply to Organizations by default — turn on this option to make the billing plans available by default to subscriber Organizations.

   Services

   - DVR depth — the number of days to store continuous recording.
   - DVR space — the volume of storage for recorded motion events, in GB.
   - LPR — license plate recognition.

> **Note**
>
> All **Services** must be filled in. For example, if you want to set up a billing plan with license plate recognition without a DVR archive, set the values of **DVR depth** and **DVR space** to 0.

After you add a billing plan, it will be displayed in the billing plans table. Also, in this table, you can see when the billing plan started working.

SETTING A NEW PRICE

In the process of providing the service to subscribers, you can change the subscription price for billing plans. To do that, follow these steps:

1. Open the price editing form by clicking on the **Edit** button.
2. In the appeared form, click **Add new price** and enter the price that will be effective starting from the current day.

4.8.4 Managing Billing Users

The next important step in setting up billing for VSAAS.IO is to add users.

Go to the **Users** section. You will see the list of all billing users.

To find the right user, use the **Search** field or the access rights filter, where values are available:

- Domain admin — administrators of your domain.
- Domain accountant — accounting of your domain.
- Orgaization admin — administrators of your subscribers.
- Organization accountant — accounting of your subscribers.

ADDING A USER

You can add two user types for your domain:

- Domain administrator. The user has all the necessary privileges to configure and administer your service.
- Domain accountant. The user can see statistics on the use of the domain by Organizations (the list of invoices to be paid, billing plans, the number of cameras that have each billing plan).

To add a user, follow these steps:

1. Click **Create new user**.
2. In the appeared form, enter the user's email address. Bind the user's email to the subscriber's Organization and the rights within the Organization by filling out the **Organization** and **Organization permission** fields.

A confirmation email will be sent to the specified email address. The user should click the link in the email and set their password.

BLOCKING A USER

If necessary, you can block the user's account. To do this, follow these steps:

1. Open the user profile in the **Users** section.
2. Select the user you want to block, and clear the **Activated** check box.

PASSWORD RECOVERY

Any user can recover a lost or forgotten password. The password can be restored either with the help of the administrator or by yourself.

To restore the password with the help of the administrator, the administrator must log in to the user's profile and click **Reset password** to send the user an email with a link to restore the password.

To restore the password themselves, the users must click the **Forgot password** button on the authorization page. The users will receive an email with a link to restore the password.

4.8.5 Managing Organizations via billing

Your cloud-based Flussonic Watcher is managed by billing, so all actions that in one way or another affect settlements with subscribers must be performed through the billing. These actions include everything related to Organizations.

To open the subscriber Organization management section, click **Organizations** in the menu. You will see the list of all Organizations of all your subscribers.

ADDING AN ORGANIZATION

To add a new organization, follow these steps:

1. Click **Add new organization**.
2. Enter the following data:
3. Title — the name of a subscriber's organization.
4. Owner — the owner of the Organization is your subscriber with whom you signed a Contract to provide cloud video surveillance services. If a user has already been created for the subscriber, you must select it from the list. If there is no user, you can create one by clicking **Add** and specifying their email address where the activation email will be sent.
5. Billing plans — the list of billing plans available for the subscriber's organization. Billing plans must be added beforehand in the billing system.
6. Trial — the period during which the billing will not calculate the amount to be paid by the subscriber.

EDITING AN ORGANIZATION

You can change the subscriber's Organization settings at any time. For example, you can specify a new Owner who will be responsible for settlements with you.

You can do this by clicking **Edit** on the **Organizations** page.

STATISTICS AND SETTLEMENTS

For each organization, you can view statistics on connected cameras and issue an invoice based on this statistics. Statistics include information about the billing plan used for each camera and the period of time during which the camera uses this billing plan.

To view statistics for the selected Organization, follow the steps below:

1. Go to the list of Organizations and click the **Usages** button.
2. In the appeared form, specify the time period for which you want to get statistics for the Organization and click **Apply**. Billing will provide detailed statistics for all cameras in the Organization and calculate the sum to be paid based on the price of the applied billing plans.

4.8.6 Integrating VSAAS.IO billing service and cloud.vsaas.io cloud video surveillance service

In this article, you will learn how a provider can implement VSAAS.IO billing service and cloud video surveillance service cloud.vsaas.io to sell camera subscriptions and do accounting for subscribers and organizations.

As a provider, do the following:

1. Sign up to VSAAS.IO account
2. See typical billing and cloud Watcher usage scenarios
    a. Create a billing plan
    b. Create an organization with a list of available billing plans
    c. Create a user with read-only permissions
    d. Create a user to edit the camera
    e. Remove the user with the permission to edit the camera
    f. Add billing plans to an existing organization
    g. Change the default billing plan
    h. Disable the camera

Glossary

- **Provider** is a customer of Flussonic that rents our billing and cloud-based video surveillance service.
- **Subscriber** is a Provider's customer, the user of the cloud.vsaas.io service.
- **Preset** is a set of DVR and analytics parameters that you can use as a template when creating and configuring cameras. Presets correspond to the billing plans. You should create and configure presets before adding cameras.
- **VSAAS.IO** (my.vsaas.io) is a billing service that manages billing plans of Provider's services to subscribers and payments for using cloud.vsaas.io. Users in VSAAS.IO are Administrators and accountants who collect information about who owes whom and how much money.
- **cloud.vsaas.io** is Flussonic Watcher's cloud-based video surveillance service. Provider's customers (organizations) use cloud.vsaas.io. It defines all technical capabilities of the video surveillance service: management of subscribers' and installers' accounts, cameras and their settings, features of billing plans.

Features of VSAAS.IO and cloud.vsaas.io

VSAAS.IO and cloud.vsaas.io are different entities. VSAAS.IO communicates with cloud.vsaas.io, but not vice versa. This means that you won't get billing data from cloud.vsaas.io. VSAAS.IO handles billing, and cloud.vsaas.io handles cameras.
VSAAS.IO and cloud.vsaas.io have their own API. Some methods are available in VSAAS.IO (my.vsaas.io) and aren't available in cloud.vsaas.io and vice versa. For example, in VSAAS.IO (my.vsaas.io) you can create or modify an organization, while in cloud.vsaas.io you can only get organization data.

Billing is calculated once a day around midnight. Do not configure the integration to regularly poll the billing system. New cameras added today will be visible tomorrow.

More details about VSAAS.IO billing here

Signing up a provider to VSAAS.IO account

An account on VSAAS.IO is needed to set up tariffs and issue invoices to subscribers. We create accounts upon request, email us specifying the name of your organization. We will provide you with a domain like `YOURNAME.vsaas.io`, or discuss the possibility of linking your own domain.

In the future, you will be able to log in directly to the personal account using the email from which you sent the request.

Typical billing and cloud Watcher usage scenarios

This section provides scenarios for using VSAAS.IO billing and cloud.vsaas.io cloud video surveillance service.

> ⚠ **Warning**
>
> my.vsaas.io and cloud.vsaas.io use the API v1 and v2 accordingly. We will migrate to API v3 to achieve a common format among all products of the Flussonic ecosystem. You can learn about these and other updates from the blog.

CREATE A BILLING PLAN

1. Log in to your Administrator account at my.vsass.io.
2. Create a billing plan by going to the **BILLING PLANS** tab and clicking **CREATE NEW BILLING PLAN**. Fill in the form and click **SAVE** to save the billing plan. The added billing plan along with its start date will be displayed in the list of billing plans under the **BILLING PLANS** tab.
3. Look up the billing plan ID by calling the GET https://my.vsaas.io/api/v1/tariff_plans method. The billing plan has two IDs: one is for my.vsaas.io— `id` and the other one is for cloud.vsaas.io— `external_id` . Use the method for mapping to CRM.

CREATE AN ORGANIZATION WITH A LIST OF AVAILABLE BILLING PLANS

1. Create an organization by calling the my.vsaas.io API method POST https://my.vsaas.io/api/v1/organizations and specify the domain owner (`owner`) with a list of required billing plans. This method creates an organization both in billing (my.vsaas.io) and cloud Watcher (cloud.vsaas.io) assigning two different organization IDs. One is for my.vsaas.io— `id` and the other one is for cloud.vsaas.io— `external_id` .

CREATE A USER WITH READ-ONLY PERMISSIONS

Using only the cloud Watcher (cloud.vsaas.io).

1. Create a user by calling the cloud.vsaas.io API method POST /vsaas/api/v2/users.
2. Connect the user to the desired organization by calling the cloud.vsaas.io API method POST /vsaas/api/v2/organizations/ORGANIZATION-ID/users. You can find out the organization ID by using the get organization list method GET /vsaas/api/v2/organizations.
3. Get the camera folder ID using the cloud.vsaas.io API method to get a list of organization folders GET vsaas/api/v2/organizations/ORGANIZATION-ID/folders.
4. Grant permissions in the folders of the desired organization to the user by calling the cloud.vsaas.io API method POST /vsaas/api/v2/organizations/ORGANIZATION-ID/folders/FOLDER-ID/users.
5. Remove the user from the default organization by calling the cloud.vsaas.io API method DELETE /vsaas/api/v2/organizations/ORGANIZATION-ID/users/USER-ID. Now the subscriber is assigned to the correct organization.

CREATE A USER TO EDIT THE CAMERA

The camera installer is the example of a user to edit the camera.

Accessing cloud Watcher (cloud.vsaas.io) only.

> **Note**
>
> The difference between creating a user for camera installation and creating a user for a subscriber with read-only permissions is that when the user connects to an organization, the user is given special permissions to add cameras (see step 2).

1. Create a user by calling the cloud.vsaas.io API method POST /vsaas/api/v2/users.
2. Connect a user with special permission to add cameras: `“can_edit_organization_cameras”: true` to the organization. To do this, call the cloud.vsaas.io API method POST /vsaas/api/v2/organizations/ORGANIZATION-ID/users. You can find out the organization ID by using the get organization list method GET /vsaas/api/v2/organizations.
3. Get the camera folder ID using the cloud.vsaas.io API method to get a list of organization folders GET vsaas/api/v2/organizations/ORGANIZATION-ID/folders.
4. Grant permissions in the folders of the desired organization to the user by calling the cloud.vsaas.io API method POST /vsaas/api/v2/organizations/ORGANIZATION-ID/folders/FOLDER-ID/users.
5. Remove the user from the default organization by calling the cloud.vsaas.io API method DELETE /vsaas/api/v2/organizations/ORGANIZATION-ID/users/USER-ID. Now the subscriber is assigned to the correct organization.

REMOVE THE USER WITH THE PERMISSION TO EDIT THE CAMERA

Remove a user with special permission to edit cameras (installer) using the cloud.vsaas.io API method DELETE '/vsaas/api/v2/users/USER-ID.

ADD BILLING PLANS TO AN EXISTING ORGANIZATION

Update the organization's billing plans using the my.vsaas.io API method PUT /organizations/ORGANIZATION-ID.

CHANGE THE DEFAULT BILLING PLAN

1. Log in to your Administrator account at my.vsass.io.
2. Create a billing plan by going to the **BILLING PLANS** tab and clicking **CREATE NEW BILLING PLAN**. Fill in the form and check *Apply to Organisations by default*.
3. Save the billing plan by clicking **SAVE**. The new billing plan along with its start date will be displayed in the list of billing plans under the **BILLING PLANS** tab.

The previous default billing plan moves to the regular billing plans because there can only be one default billing plan.

DISABLE THE CAMERA

There are two ways to do this:

- Using the VSAAS.IO web interface.
  a. Log in to the Administrator account in cloud.vsaas.io.
  b. Move to the **My cameras** tab.
  c. Click the settings of the desired camera (gear icon), find the **SETTING OPTIONS**, and uncheck the **Enabled** checkbox.
  d. Save the settings by clicking **Save** or pressing *Ctrl+S*.
- Using the cloud.vsaas.io API.

  Use the PUT vsaas/api/v2/cameras/OBJECT-ID method.

VSAAS.IO API

API v1 for billing VSAAS.IO.

AUTHORIZATION

Only the domain owner and Administrator can make requests to the VSAAS.IO API.

All requests to VSAAS.IO API require authorization with:

- username (`X-USER: ADMIN-EMAIL-ADDRESS`),
- password (`X-PASSWORD: ADMIN-PASSWORD`),
- domain ID (`X-DOMAIN-ID: 000`). Ask your manager for the domain ID.

BILLING PLANS

Managing billing plans in my.vsaas.io.

Get a list of all available billing plans

curl -H "Content-Type: application/json" \ -H "X-USER: ADMIN-EMAIL-ADDRESS" \ -H "X-PASSWORD: ADMIN-PASSWORD" \ -H "X-DOMAIN-ID: 000" \ -X GET 'https://my.vsaas.io/api/v1/tariff_plans'

Response example:

{ "tariff_plans": [{ "id": 549, "name": "Test plan", "date_from": "27/04/2024", "full_price": "100.0 USD", "status": "Active", "external_id": 392 }, { "id": 548, "name": "Default", "date_from": "27/04/2024", "full_price": "0.0 USD", "status": "Active", "external_id": 391 }] }

ORGANIZATIONS

Managing organizations in my.vsaas.io.

Get a list of all available organizations or data on a specific organization

curl -H "Content-Type: application/json" \ -H "X-USER: ADMIN-EMAIL-ADDRESS" \ -H "X-PASSWORD: ADMIN-PASSWORD" \ -H "X-DOMAIN-ID: 000" \ -X GET 'https://my.vsaas.io/api/v1/organizations'

Response example:

{ "organizations": [{ "id": 736, "title": "My organization", "domain": { "id": 247, "title": "YOUR-DOMAIN-NAME" }, "owner": { "id": 410, "email": "demo@example.org" }, "discount": "0%", "trial_expired_at": "", "external_id": 753, "tariff_plans": [{ "id": 548, "name": "Default", "date_from": "27/04/2024", "full_price": "0.0 USD", "status": "Active", "external_id": 391 }] }] }

Add the organization with billing plans or without them

This call creates an organization in both VSAAS.IO and cloud.vsaas.io with a default billing plan `Default` .

curl -H "Content-Type: application/json" \ -H "X-USER: ADMIN-EMAIL-ADDRESS" \ -H "X-PASSWORD: ADMIN-PASSWORD" \ -H "X-DOMAIN-ID: 000" \ -X POST 'https://my.vsaas.io/api/v1/organizations' -d '{"organization": {"title": "Demo org", "owner": "demo@example.org", "tariff_plans": [549]}}'

where:

- (required) `title` is the name of the organization.
- (required) `owner` is the e-mail address of the owner of the organization. It can be either existing or arbitrary. If you specify an arbitrary address, a new user will be created and an e-mail will be sent to the specified address with a request to confirm it and come up with the password.
- `tariff_plans` is the list of billing plan IDs. You can find out the billing plan ID by getting a list of all available billing plans.

Response example:

{ "organization": { "id": 797, "title": "Demo org", "domain": { "id": 247, "title": "YOUR-DOMAIN-NAME" }, "owner": { "id": 410, "email": "demo@example.org" }, "discount": "0%", "trial_expired_at": "", "external_id": 774, "tariff_plans": [{ "id": 548, "name": "Default", "date_from": "27/04/2024", "full_price": "0.0 USD", "status": "Active", "external_id": 391 }, { "id": 549, "name": "Test plan", "date_from": "27/04/2024", "full_price": "100.0 USD", "status": "Active", "external_id": 392 }] } }

Modify billing plans of the existing organization

curl -H "Content-Type: application/json" \ -H "X-USER: ADMIN-EMAIL-ADDRESS" \ -H "X-PASSWORD: ADMIN-PASSWORD" \ -H "X-DOMAIN-ID: 000" \ -X PUT 'https://my.vsaas.io/api/v1/organizations/ORGANIZATION-ID' -d '{"organization": {"tariff_plans": [550]}}'

- (required) `ORGANIZATION-ID` is the organization ID.
- `tariff_plans` is the list of billing plan IDs. You can find out the billing plan ID by getting a list of all available billing plans. If you pass an empty list, the organization will have no billing plans.

Response example:

{ "organization": { "id": 797, "title": "Unedited org", "domain": { "id": 247, "title": "bikbaeva" }, "owner": { "id": 410, "email": "demo@example.org" }, "discount": "0%", "trial_expired_at": "", "external_id": 774, "tariff_plans": [{ "id": 548, "name": "Default", "date_from": "27/04/2024", "full_price": "0.0 USD", "status": "Active", "external_id": 391 }, { "id": 550, "name": "My demo plan", "date_from": "02/05/2024", "full_price": "1500.0 USD", "status": "Active", "external_id": 393 }] } }

Remove the existing organization

```
curl -H "Content-Type: application/json" \ -H "X-USER: ADMIN-EMAIL-ADDRESS" \ -H "X-PASSWORD: ADMIN-PASSWORD" \ -H "X-DOMAIN-ID: 000" \ -X DELETE 'https://my.vsaas.io/api/v1/organizations/ORGANIZATION-ID'
```

where `ORGANIZATION-ID` is the organization ID.

Response: 204 No content

cloud.vsaas.io API

API v2 fo cloud video surveillance service cloud.vsaas.io.

In cloud.vsaas.io you can set the following permissions for users:

- edit organization cameras камеры организации (`"can_edit_organization_cameras"`)
- edit organization users (`"can_edit_organization_users"`)
- edit organizations (`"can_edit_organizations"`)
- view cameras
- view statistics on cameras (`"can_view_organization_stats"`)

AUTHORIZATION

Only the domain owner and Administrator can make requests to the VSAAS.IO API.

All requests to the cloud.vsaas.io API require authorization by the Administrator API key (`x-vsaas-api-key: VSAAS-ADMIN-API-KEY`). You can copy the Administrator API key from the cloud.vsaas.io web interface: **Admin** > **Settings** > **General Settings** > **API Key**.

CAMERAS

Managing cameras in cloud.vsaas.io.

Get a list of all cameras and their data in the domain

```
curl -H "x-vsaas-api-key: VSAAS-USER-API-KEY" \ -X GET 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/cameras'
```

where `YOUR-DOMAIN-NAME` is the domain name.

Response example:

[{ "name": "camera02-5ca25a2330", "onvif_ptz": false, "organization_id": 752, "stream_url": "fake://fake", "substream_url": "", "title": "Camera02", "user_attributes": { "favorite": false, "motion_alarm": false }, "video_only": false, "transcode_audio": false "comment": "", "coordinates": "", "enabled": false, "folder_id": 1155, "has_actions": false, "last_change": { "user": "demo@example.org", "domain_id": 143, "object_id": "camera02-5ca25a2330", "created_at": 1715105180, "event_data": { "changed_data": { "_enabled": [true, false] } }, "action_type": "change", "object_type": "Camera", "request_data": { "ip": "188.170.86.201,188.170.86.201", "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.0 Safari/537.36" } }, "motion_detector_enabled": false, "permissions": { "view": true, "edit": true, "ptz": true, "dvr": true, "dvr_depth_limit": null, "actions": true }, "postal_address": "", "precise_thumbnails_days": 0, "preset": { "domain_id": 143, "dvr_depth": 0.0, "id": 391, "is_adjustable": false, "is_default": true, "is_deleted": false, "precise_thumbnails_days": 0, "title": "Default", "vision_params": {} }, "preset_id": 391, "static": true, "stream_status": { "name": "camera02-5ca25a2330", "server": null, "alive": false, "lifetime": null, "bitrate": null, "source_error": "no_stream_status", "http_port": null, "https_port": null } }, ...]

> **Note**
>
> The response returns the ID of the preset, but it does not return the ID of the billing plan defined in the billing system. This means that you need to compare the preset ID with the billing plan ID on your side.

Get the data on the camera

```
curl -H "x-vsaas-api-key: VSAAS-USER-API-KEY" \ -X GET 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/cameras/OBJECT-ID'
```

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `OBJECT-ID` is the camera ID. Find out the camera ID using GET /vsaas/api/v2/cameras or GET /vsaas/api/v2/cameras?organization_id=ORGANIZATION-ID.

Response will be same as on cameras list, but contains only a one element.

Get a list of all cameras and their data in the organization

```
curl -H "x-vsaas-api-key: VSAAS-USER-API-KEY" \ -X GET 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/cameras?organization_id=ORGANIZATION-ID'
```

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `ORGANIZATION-ID` is the organization ID assigned when creating the organization.

Response example:

[{ "comment": "", "coordinates": "", "enabled": false, "folder_id": 1155, "groups": [], "has_actions": false, "last_change": { "user": "a.bikbaeva@erlyvideo.org", "domain_id": 143, "object_id": "camera02-5ca25a2330", "created_at": 1715105180, "event_data": { "changed_data": { "_enabled": [true, false], "extra_config": [null, { "video_only": false, "transcode_audio": false }] } }, "action_type": "change", "object_repr": "", "object_type": "Camera", "request_data": { "ip": "188.170.86.201,188.170.86.201", "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.0 Safari/537.36" } }, "motion_detector_enabled": false, "name": "camera02-5ca25a2330", "onvif_ptz": false, "organization_id": 752, "permissions": { "view": true, "edit": true, "ptz": true, "dvr": true, "dvr_depth_limit": null, "actions": true }, "playback_config": { "token": "3.sJXDwZNHAAAAAAAAAAAAAAAAAAAAAjylfLugHFgvX5iaFryiQBGFeBkLA" }, "postal_address": "", "precise_thumbnails_days": 0, "preset": { "domain_id": 143, "dvr_depth": 0.0, "id": 391, "is_adjustable": false, "is_default": true, "is_deleted": false, "precise_thumbnails_days": 0, "title": "Default", "vision_params": {} }, "preset_id": 391, "static": true, "stream_status": { "name": "camera02-5ca25a2330", "server": null, "alive": false, "lifetime": null, "bitrate": null, "source_error": "no_stream_status", "http_port": null, "https_port": null }, "stream_url": "fake://fake", "provision_required": false, "substream_url": "", "thumbnails": false, "title": "Camera02", "user_attributes": { "favorite": false, "motion_alarm": false }, "video_only": false, "transcode_audio": false }, { "comment": "", "coordinates": "", "enabled": true, "folder_id": 1155, "groups": [], "has_actions": false, "last_change": { "user": "a.bikbaeva@erlyvideo.org", "domain_id": 143, "object_id": "camera05-6f057b160a", "created_at": 1714737887, "event_data": { "changed_data": { "extra_config": [null, { "video_only": false, "transcode_audio": false }] } }, "action_type": "change", "object_repr": "", "object_type": "Camera", "request_data": { "ip": "188.242.170.19,188.242.170.19", "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.0 Safari/537.36" } }, "motion_detector_enabled": false, "name": "camera05-6f057b160a", "onvif_ptz": false, "organization_id": 752, "permissions": { "view": true, "edit": true, "ptz": true, "dvr": true, "dvr_depth_limit": null, "actions": true }, "playback_config": { "token": "3.sJXDwZNHAAAAAAAAAAAAAAAAAAAAAj9odoI-86sxLhc4uyZT9PcR5TedY" }, "postal_address": "", "precise_thumbnails_days": 0, "preset": { "domain_id": 143, "dvr_depth": 0.0, "id": 391, "is_adjustable": false, "is_default": true, "is_deleted": false, "precise_thumbnails_days": 0, "title": "Default", "vision_params": {} }, "preset_id": 391, "static": true, "stream_status": { "name": "camera05-6f057b160a", "server": "eu-004.vscdn.io", "alive": true, "lifetime": 385264839, "bitrate": 196, "source_error": null, "http_port": null, "https_port": null }, "stream_url": "fake://fake", "streamer_hostname": "eu-004.vscdn.io", "provision_required": false, "substream_url": "", "thumbnails": false, "title": "Camera05", "user_attributes": { "favorite": false, "motion_alarm": false }, "video_only": false, "transcode_audio": false }]

Change the billing plan for the camera

```
curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -H 'Content-Type: application/json' \ -X PUT 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/cameras/OBJECT-ID' --data '{"preset_id": 526}'
```

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `OBJECT-ID` is the camera ID. Find out the camera ID using GET /vsaas/api/v2/cameras or GET /vsaas/api/v2/cameras?organization_id=ORGANIZATION-ID.
- `preset_id` is the ID of the preset you want to set for the camera. Find out the preset ID using GET /vsaas/api/v2/cameras?organization_id=ORGANIZATION-ID.

Response example:

{ "comment": "", "coordinates": "", "enabled": true, "folder_id": 1179, "groups": [], "has_actions": false, "last_change": { "user": "domain_143_via_apikey", "domain_id": 143, "object_id": "cam-10-9bdbc979dd", "created_at": 1715128188, "event_data": { "changed_data": { "preset": ["", ""], "dvr_depth": [3.0, null], "dvr_space": [6, null], "preset_id": [393, 391] } }, "action_type": "change", "object_repr": "", "object_type": "Camera", "request_data": { "ip": "188.242.170.19,188.242.170.19", "user_agent": "PostmanRuntime/7.32.1" } }, "motion_detector_enabled": false, "name": "cam-10-9bdbc979dd", "onvif_ptz": false, "organization_id": 774, "permissions": { "view": true, "edit": true, "ptz": true, "dvr": true, "dvr_depth_limit": null, "actions": true }, "playback_config": { "token": "3.sJXDwZNHAAAAAAAAAAAAAAAAAAAj2eRzt7bH5sYfDpRVJ4jlfvwAO4o" }, "postal_address": "", "precise_thumbnails_days": 0, "preset": { "domain_id": 143, "dvr_depth": 0.0, "id": 391, "is_adjustable": false, "is_default": true, "is_deleted": false, "precise_thumbnails_days": 0, "title": "Default", "vision_params": {} }, "preset_id": 391, "static": true, "stream_status": { "name": "cam-10-9bdbc979dd", "server": "eu-004.vscdn.io", "alive": true, "lifetime": 730640, "bitrate": 194, "source_error": null, "http_port": null, "https_port": null }, "stream_url": "fake://fake", "streamer_hostname": "eu-004.vscdn.io", "provision_required": false, "substream_url": "", "thumbnails": false, "title": "cam-10", "user_attributes": { "favorite": false, "motion_alarm": false }, "video_only": false, "transcode_audio": false }

Disable the camera

```
curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -H 'Content-Type: application/json' \ -X PUT 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/cameras/OBJECT-ID' --data '{"enabled": false}'
```

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `OBJECT-ID` is the camera ID. Find out the camera ID using GET /vsaas/api/v2/cameras or GET /vsaas/api/v2/cameras?organization_id=ORGANIZATION-ID.

Response example:

{ "comment": "", "coordinates": "", "enabled": false, "folder_id": 1156, "groups": [], "has_actions": false, "last_change": { "user": "domain_143_via_apikey", "domain_id": 143, "object_id": "demo-cam-219c36899q", "created_at": 1715103531, "event_data": { "changed_data": { "_enabled": [true, false] } }, "action_type": "change", "object_repr": "", "object_type": "Camera", "request_data": { "ip": "188.242.169.75,188.242.169.75", "user_agent": "PostmanRuntime/7.32.1" } }, "motion_detector_enabled": false, "name": "demo-cam-219c36899q", "onvif_ptz": false, "organization_id": 753, "permissions": { "view": true, "edit": true, "ptz": true, "dvr": true, "dvr_depth_limit": null, "actions": true }, "playback_config": { "token": "3.sJXDwZNHAAAAAAAAAAAAAAAAAAAj5fUIZsybhtcpRTPPwzvGGht3iDs" }, "postal_address": "", "precise_thumbnails_days": 0, "preset": { "domain_id": 143, "dvr_depth": 0.0, "id": 391, "is_adjustable": false, "is_default": true, "is_deleted": false, "precise_thumbnails_days": 0, "title": "Default", "vision_params": {} }, "preset_id": 391, "static": true, "stream_status": { "name": "demo-cam-219c36899q", "server": null, "alive": false, "lifetime": null, "bitrate": null, "source_error": "no_stream_status", "http_port": null, "https_port": null }, "stream_url": "fake://fake", "streamer_hostname": "eu-004.vscdn.io", "provision_required": false, "substream_url": "", "thumbnails": false, "title": "demo-cam", "user_attributes": { "favorite": false, "motion_alarm": false }, "video_only": false, "transcode_audio": false }

Enable the camera

```
curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -H 'Content-Type: application/json' \ -X PUT 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/cameras/OBJECT-ID' --data '{"enabled": true}'
```

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `OBJECT-ID` is the camera ID. Find out the camera ID using GET /vsaas/api/v2/cameras or GET /vsaas/api/v2/cameras?organization_id=ORGANIZATION-ID.

Response example:

```
{ "comment": "", "coordinates": "", "enabled": true, "folder_id": 1156, "groups": [], "has_actions": false, "last_change": { "user": "domain_143_via_apikey", "domain_id": 143, "object_id": "demo-cam-219c36899q", "created_at": 1715128546, "event_data": { "changed_data": { "_enabled": [ false, true ] } }, "action_type": "change", "object_repr": "", "object_type": "Camera", "request_data": { "ip": "188.242.170.19,188.242.170.19", "user_agent": "PostmanRuntime/7.32.1" } }, "motion_detector_enabled": false, "name": "demo-cam-219c36899q", "onvif_ptz": false, "organization_id": 753, "permissions": { "view": true, "edit": true, "ptz": true, "dvr": true, "dvr_depth_limit": null, "actions": true }, "playback_config": { "token": "3.sJXDwZNHAAAAAAAAAAAAAAAAAAAAj5fUIZsqbhtcpRTPPszvGGht3iDs" }, "postal_address": "", "precise_thumbnails_days": 0, "preset": { "domain_id": 143, "dvr_depth": 0.0, "id": 391, "is_adjustable": false, "is_default": true, "is_deleted": false, "precise_thumbnails_days": 0, "title": "Default", "vision_params": {} }, "preset_id": 391, "static": true, "stream_status": { "name": "demo-cam-219c36899q", "server": null, "alive": true, "lifetime": 145967640, "bitrate": 202, "source_error": null, "http_port": null, "https_port": null }, "stream_url": "fake://fake", "provision_required": false, "substream_url": "", "thumbnails": false, "title": "demo-cam", "user_attributes": { "favorite": false, "motion_alarm": false }, "video_only": false, "transcode_audio": false }
```

USERS

Managing users in cloud.vsaas.io.

Get a list of all users

```
curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -X GET 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/users'
```

where `YOUR-DOMAIN-NAME` is the domain name.

Response example:

```
[ { "id": 1297, "domain_id": 111, "login": "demo@example.org", "apikey": "ajG3NIzOt2CjUyRoWyVeh2U9", "is_admin": false, "authorized_ip": null, "enabled": true, "readonly": false, "monitoring": false, "name": null, "locale": "", "note": null, "phone": null, "notification_email": null, "external_id": null, "max_sessions": null, "can_edit_organizations": false, "can_edit_settings": true, "can_view_organizations": true, "organizations": [ 752, 753 ] }, { "id": 1298, "domain_id": 111, "login": "another-demo@example.org", "apikey": null, "is_admin": false, "authorized_ip": null, "enabled": true, "readonly": false, "monitoring": false, "name": "", "locale": "en", "note": "", "phone": null, "notification_email": null, "external_id": null, "max_sessions": null, "can_edit_organizations": false, "can_edit_settings": false, "can_view_organizations": false, "organizations": [ 753, 762 ] }, ...
```

Create a user

```
curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -H 'Content-Type: application/json' \ -X POST 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/users' --data-raw '{ "login": "user@example.com", "password": "qwe123" } '
```

where:

- `login` is the unique username.
- `password` is the password for the user.

Response example:

```
{ "id": 1377, "domain_id": 143, "login": "user@example.com", "apikey": null, "is_admin": false, "authorized_ip": null, "enabled": true, "readonly": false, "monitoring": false, "name": null, "locale": "", "note": null, "phone": null, "notification_email": null, "external_id": null, "max_sessions": null, "can_edit_organizations": false, "can_edit_settings": false, "can_view_organizations": false, "organizations": [ 752 ] }
```

The new user is automatically assigned to the default `Unassigned cameras` organization.

Remove a user

```
curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -X DELETE 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/users/USER-ID'
```

where `YOUR-DOMAIN-NAME` is the domain name.

Response:

```
{ "success": true }
```

ORGANIZATIONS

Managing organizations in cloud.vsaas.io.

Get a list of all organizations

```
curl -H "x-vsaas-api-key: VSAAS-USER-API-KEY" \ -X GET 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/organizations'
```

where `YOUR-DOMAIN-NAME` is the domain name.

Response example:

```
[ { "domain": { "title": "YOUR-DOMAIN-NAME", "id": 143 }, "title": "My organization", "mosaic_count": 0, "camera_count": 4, "owner": { "login": "demo@example.org", "id": 1179 }, "can_view_stats": true, "can_edit_cameras": true, "camera_limit": 1000, "dvr_limit": 100, "can_edit_users": true, "id": 753, "user_count": 2, "is_member": false, "user_limit": 1000, "is_default": false }, { "domain": { "title": "YOUR-DOMAIN-NAME", "id": 143 }, "title": "Unassigned cameras", "mosaic_count": 0, "camera_count": 2, "owner": { "login": "demo@example.org", "id": 1179 }, "can_view_stats": true, "can_edit_cameras": true, "camera_limit": 1000, "dvr_limit": 100, "can_edit_users": true, "id": 752, "user_count": 1, "is_member": false, "user_limit": 1000, "is_default": true } ]
```

Get a list of all available billing plans for the organization

```
curl -H "x-vsaas-api-key: VSAAS-USER-API-KEY" \ -X GET 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/organizations/ORGANIZATION-ID/presets'
```

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `ORGANIZATION-ID` is the organization ID assigned when creating the organization.

Response example:

```
[ { "organization_id": null, "preset": { "dvr_depth": 0.0, "dvr_lock_days": null, "vision_params": {}, "vision_gpu": null, "vision_alg": null, "is_deleted": false, "vision_enabled": null, "is_adjustable": false, "precise_thumbnails_days": 0, "id": 391, "title": "Default", "dvr_space": null, "vision_areas": null }, "domain_id": 143, "id": null, "preset_id": 391 } ]
```

USERS IN ORGANIZATION

Managing users in organization in cloud.vsaas.io.

Get a list of all users of the organization

```
curl -H "x-vsaas-api-key: VSAAS-USER-API-KEY" \ -X GET 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/organizations/ORGANIZATION-ID/users'
```

where:

- `YOUR-DOMAIN-NAME` is the domain name
- `ORGANIZATION-ID` is the organization ID assigned when creating the organization.

Response example:

```
[ { "notification_email": null, "can_view_organization_stats": true, "can_view_organization_plists": true, "login": "demo@example.org", "can_edit_organization_cameras": true, "can_edit_organization_plists": true, "user_id": 1297, "folders_permissions": [], "can_edit_organization_users": true }, { "notification_email": null, "can_view_organization_stats": false, "can_view_organization_plists": false, "login": "demo-2@example.com", "can_edit_organization_cameras": false, "can_edit_organization_plists": false, "user_id": 1377, "folders_permissions": [], "can_edit_organization_users": false } ]
```

Assign the user with read-only permissions to the organization

```
curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -H 'Content-Type: application/json' \ -X POST 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/organizations/ORGANIZATION-ID/users' --data '{ "user_id": 1378 }'
```

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `ORGANIZATION-ID` is the organization ID assigned when creating the organization.
- `user_id` is the user ID assigned when creating the user.

Response example:

{ "notification_email": null, "can_view_organization_stats": false, "can_view_organization_plists": false, "login": "viewer@example.com", "can_edit_organization_cameras": false, "can_edit_organization_plists": false, "user_id": 1378, "folders_permissions": [], "can_edit_organization_users": false }

The user has no permissions, not even the permission to view cameras: `"folders_permissions": []`.

Assign the user with permissions to edit the camera to the organization

curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -H 'Content-Type: application/json' \ -X POST 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/organizations/ORGANIZATION-ID/users' --data '{ "user_id": 1377, "can_edit_organization_cameras": true }'

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `ORGANIZATION-ID` is the organization ID assigned when creating the organization.
- `user_id` is the user ID assigned when creating the user.

Response example:

{ "notification_email": null, "can_view_organization_stats": false, "can_view_organization_plists": true, "login": "installer@example.com", "can_edit_organization_cameras": true, "can_edit_organization_plists": true, "user_id": 1377, "folders_permissions": [], "can_edit_organization_users": false }

The user doesn't have the permission to view cameras: `"folders_permissions": []`.

Remove users from the default organization

curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -X DELETE 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/organizations/ORGANIZATION-ID//users/USER-ID'

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `ORGANIZATION-ID` is the organization ID assigned when creating the organization.
- `USER-ID` is the user ID assigned when creating the user.

Response:

{ "success": true }

FOLDERS IN ORGANIZATIONS

Managing folders in organizations in cloud.vsaas.io.

Get a list of all folders of the organization

All cameras belong to some folder. That is, there are no cameras outside of folders. When you create an organization, the default folder `Cameras` is created.

curl -H "x-vsaas-api-key: VSAAS-USER-API-KEY" \ -X GET 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/organizations/ORGANIZATION-ID/folders'

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `ORGANIZATION-ID` is the organization ID assigned when creating the organization.

Response example:

[{ "level": 0, "camera_count": 4, "organization_id": 753, "parent_id": null, "map_topleft": null, "id": 1156, "title": "Cameras", "map_bottomleft": null, "order_num": 1, "map_topright": null, "coordinates": null }]

Grant permissions for users in a specific folder in the organization

```
curl -H "x-vsaas-api-key: VSAAS-ADMIN-API-KEY" \ -H 'Content-Type: application/json' \ -X POST 'https://YOUR-DOMAIN-NAME.vsaas.io/vsaas/api/v2/organizations/ORGANIZATION-ID/folders/FOLDER-ID/users' --data '{ "user_id": 1378, "can_view_dvr": 1, "can_use_ptz": 1, "can_use_actions": 1, "dvr_depth_limit": null }'
```

where:

- `YOUR-DOMAIN-NAME` is the domain name.
- `ORGANIZATION-ID` is the organization ID assigned when creating the organization.
- `FOLDER-ID` is the ID of the folder with cameras.

Response example:

```
{ "can_use_ptz": true, "can_use_actions": true, "user_id": 1378, "can_view_dvr": true, "dvr_depth_limit": null }
```

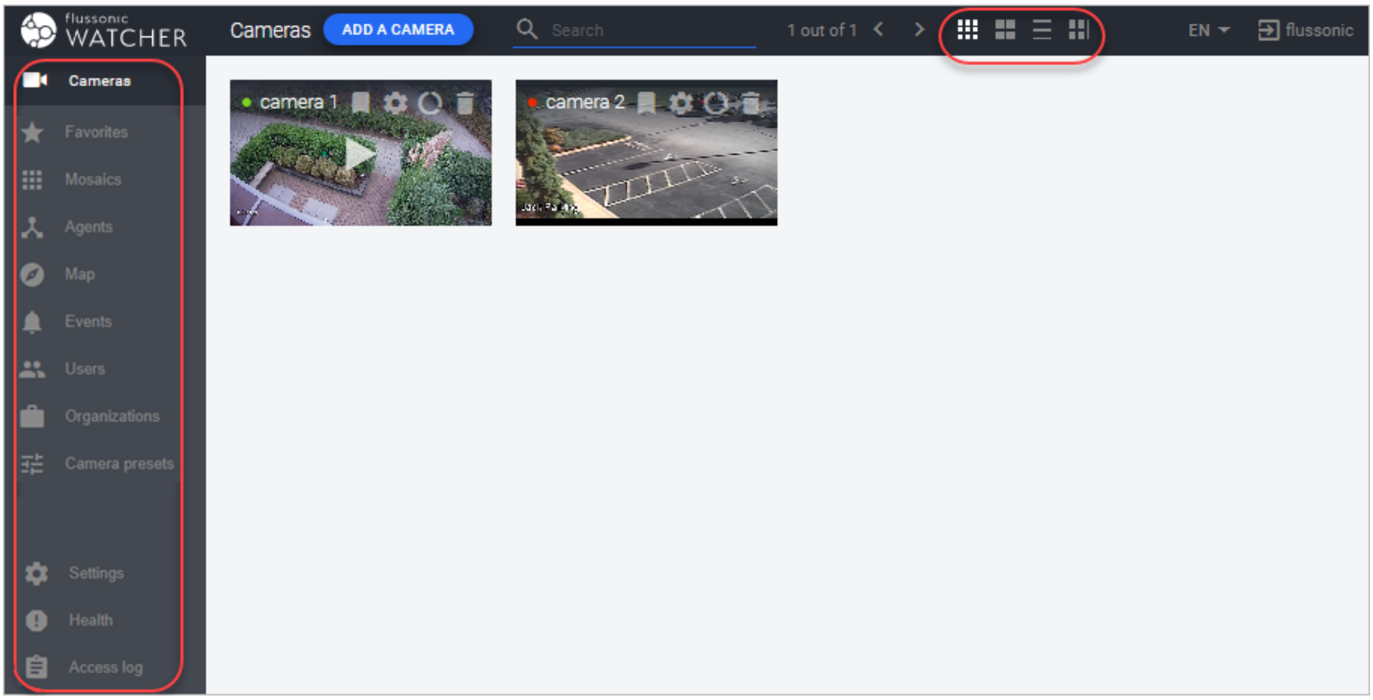
5. User Guides

5.1 Watcher user interface

Users can log into FlussoniC Watcher using their login and password after the administrator has added them to the system. They can access cameras that were assigned to them by an administrator.

To open the Watcher UI, open this URL in the browser:

```
http://MANAGING_FLUSSONIC_SERVER:80
```



5.1.1 Main menu

The Watcher UI consists of the following sections:

- Cameras. This is the main page of the Watcher interface. Here the dashboard is located where you can view and manage cameras; you can also set up cameras here.
- Favorites. Here you can access selected cameras.
- Mosaics. Mosaics are groups of cameras that are displayed together on a single page.
- Agents. Here you manage the list of cameras with Flussonic Agent installed.
- Map. You can access cameras by clicking them right on the map. The map shows only cameras with configured coordinates.
- Events. Notifications on motion, LPR or face recognition.

The following items are available for Organization's administrators (owners) only:

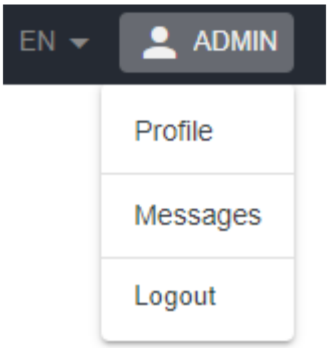
- Users. The place where you can view and manage users.
- Organizations. The place where you can view and manage Organizations and Organization users.

The following items are available for Watcher Administrators only:

- Camera presets. Camera preset management.
- Settings. Watcher settings.
- **Health**. Server health info.
- **Access log**. The history of user sessions.

5.1.2 User menu

The user menu in the top right corner provides the following items:



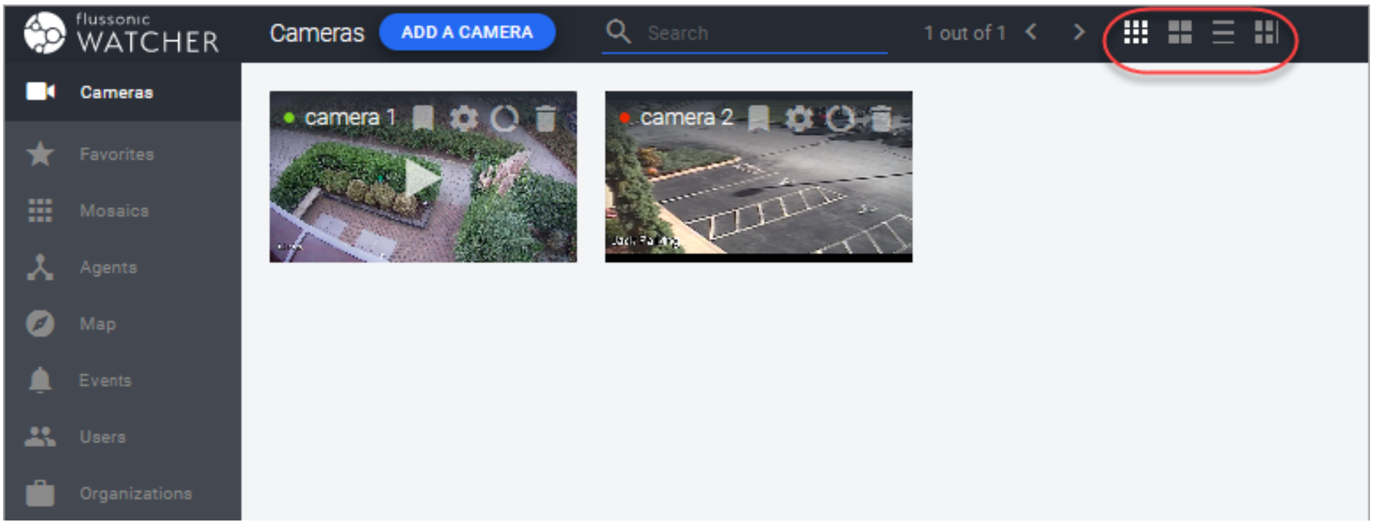
- Profile. Any user can edit the settings necessary for his work with Watcher, for example, change the username, password, email, etc.
- **Messages**. This shows messages sent to the current user by other users or administrators. Only users with access to the user list can send messages.
- **Logout**. End the session of the current user.

5.2 My Cameras

Users can view and manage cameras in the **Cameras** section (if they have permissions to cameras).

5.2.1 Camera list views

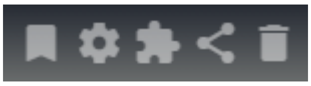
You can view the list of cameras in several modes. Click the icon in the top right corner of the **Cameras** page to select the mode:



Small cards and Cards

These are the list of automatically updated screenshots from all cameras.

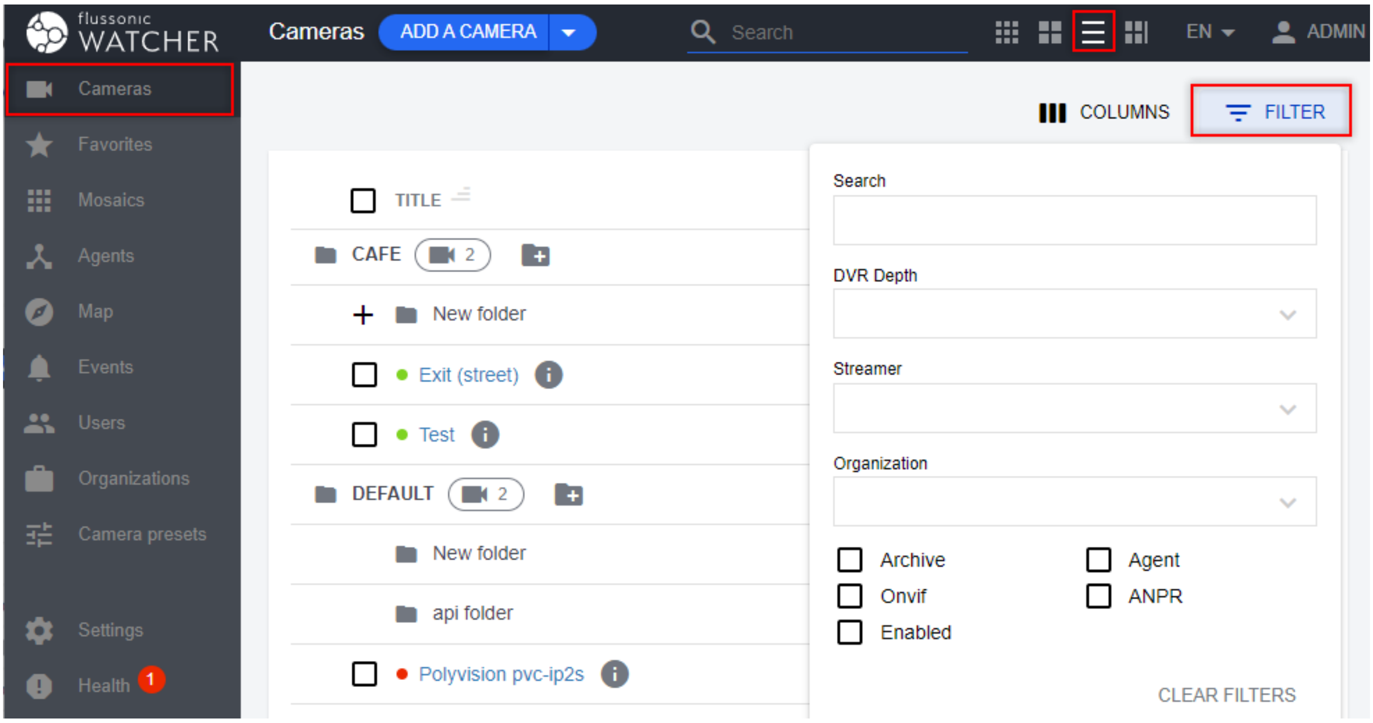
On the screenshot of each camera there are buttons that allows you to manage a camera:



- Add the camera to favorites (will appear in **Favorites**)
- Edit camera settings
- Run or add an action (any API call)
- Share the camera
- Delete the camera

List

In this mode, you will see a hierarchy of Organizations, folders and cameras.

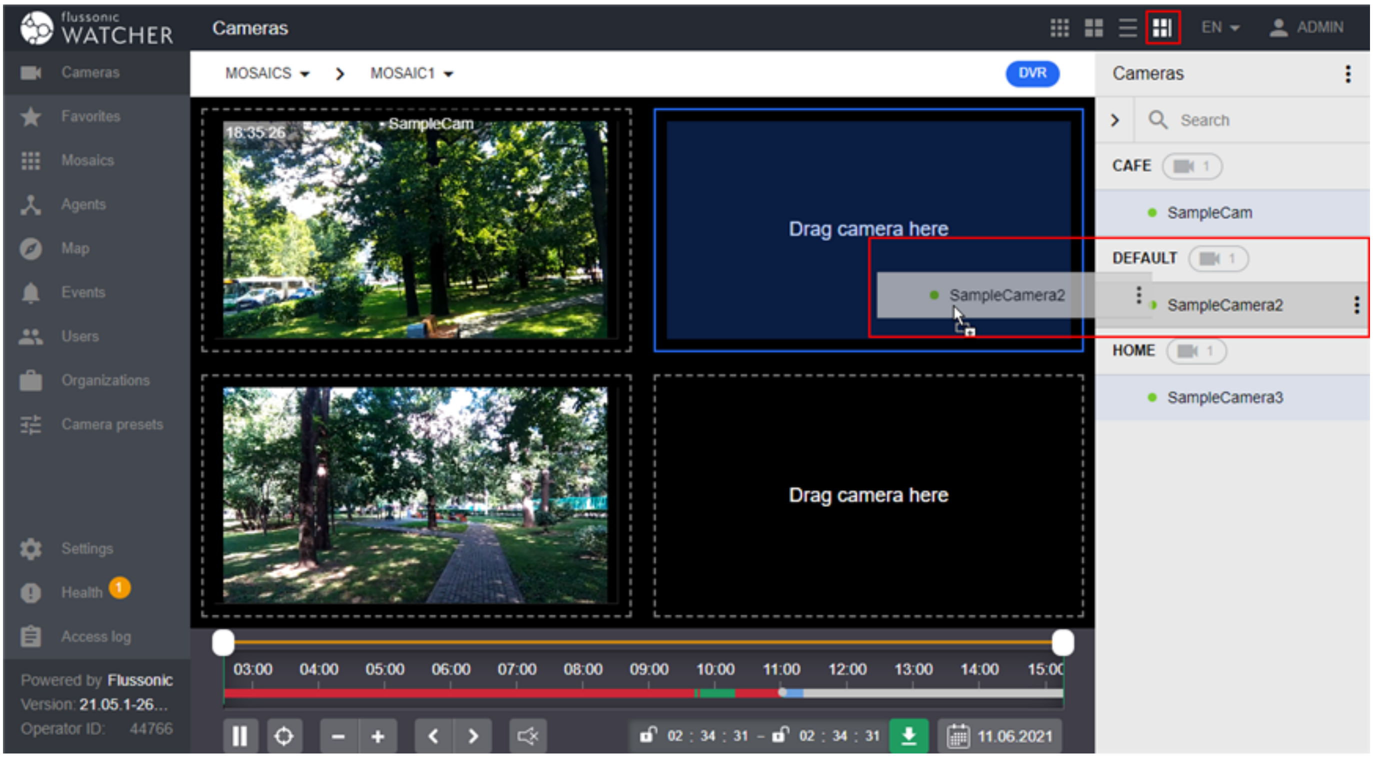


In the List mode the following *filters* are available:

- by the title
- by the depth of the DVR archive
- by the streamer where a camera is connected
- by various attributes: Archive, Agent, Onvif, ANPR (LPR, license plates recognition), Enabled (only cameras that are enabled).

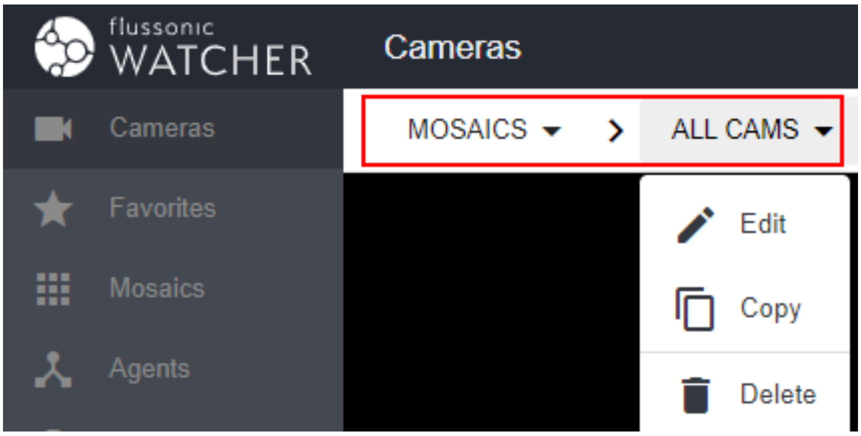
Dashboard

In this mode you can drag cameras to a previously created mosaic and immediately see the resulting mosaic. This mode is a convenient alternative to managing mosaics in the **Mosaics** or **Organizations** sections. Your changes are automatically saved.



A distinctive feature of this method of working with mosaics is that cameras from different organizations can be added to the mosaic. If users have access to such a mosaic but do not have access to some cameras in it, then they will only see their own cameras while forbidden cameras will be blackened.

Select the mosaic in the **MOSAIC** menu to view it. Use the mosaic's menu to edit, copy or delete the mosaic.

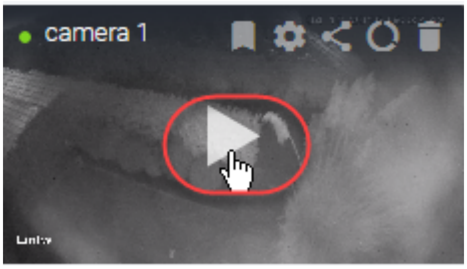


5.2.2 Viewing the camera's live video or DVR archive

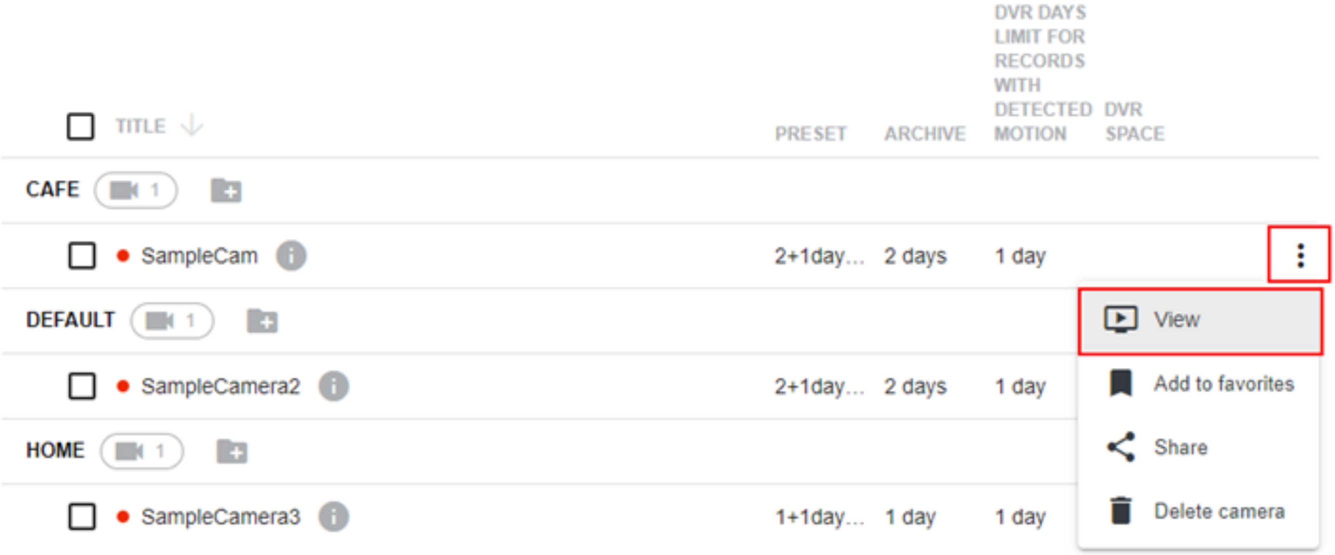
Go to **Cameras** to view live video or archive. You can access live video or archive in different ways depending on the display mode.

Video in Cards or List modes

- In **Cards** mode click the camera name or the Play button on the picture from the camera.

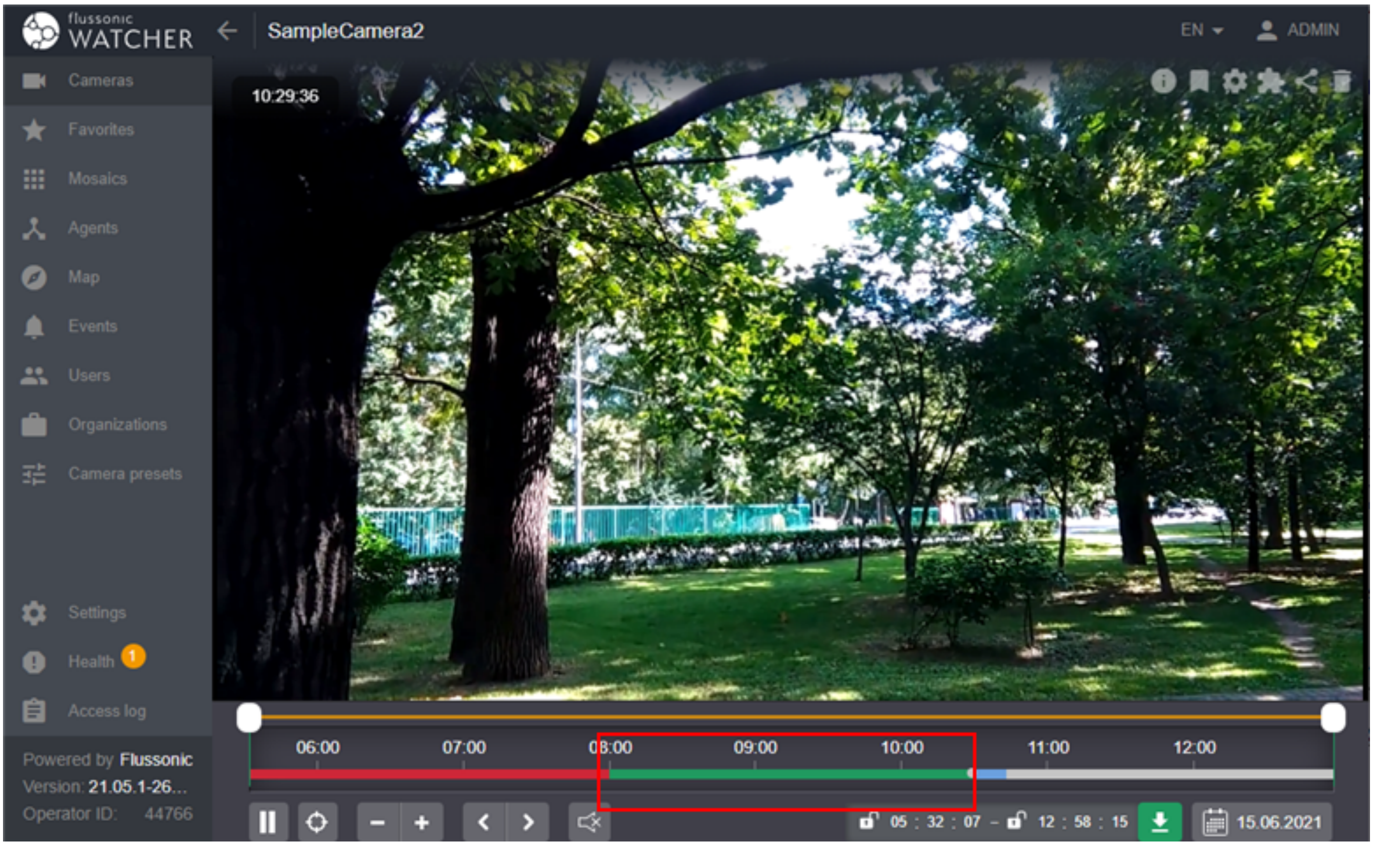


- In the **List** mode click three-dot menu then select **View**.



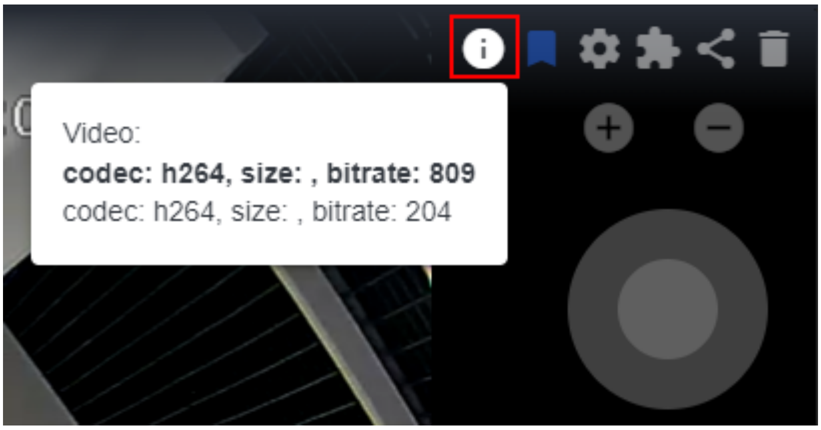
☐ TITLE	PRESET	ARCHIVE	DVR DAYS LIMIT FOR RECORDS WITH DETECTED MOTION	DVR SPACE
CAFE 1				
☐ SampleCam	2+1day...	2 days	1 day	
DEFAULT 1				
☐ SampleCamera2	2+1day...	2 days	1 day	
HOME 1				
☐ SampleCamera3	1+1day...	1 day	1 day	

Video is displayed in the player. If the camera has DVR, the player will show it as the green part of the timeline.



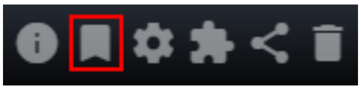
The control buttons in the top right of the video allow the following actions:

- View information about camera streams: bitrate, resolution, codec. An active stream data is highlighted in bold. Hover over the icon to see this info.

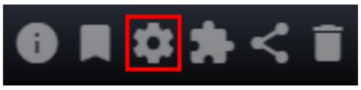


Video:
codec: h264, size: , bitrate: 809
codec: h264, size: , bitrate: 204

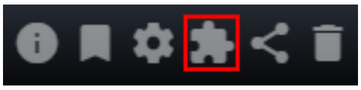
- Add the camera to favorites (will appear in **Favorites**)



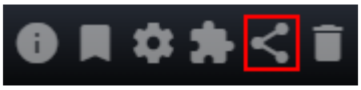
- Edit camera settings



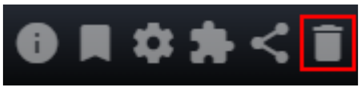
- Run or add an action (any API call)



- Share the camera

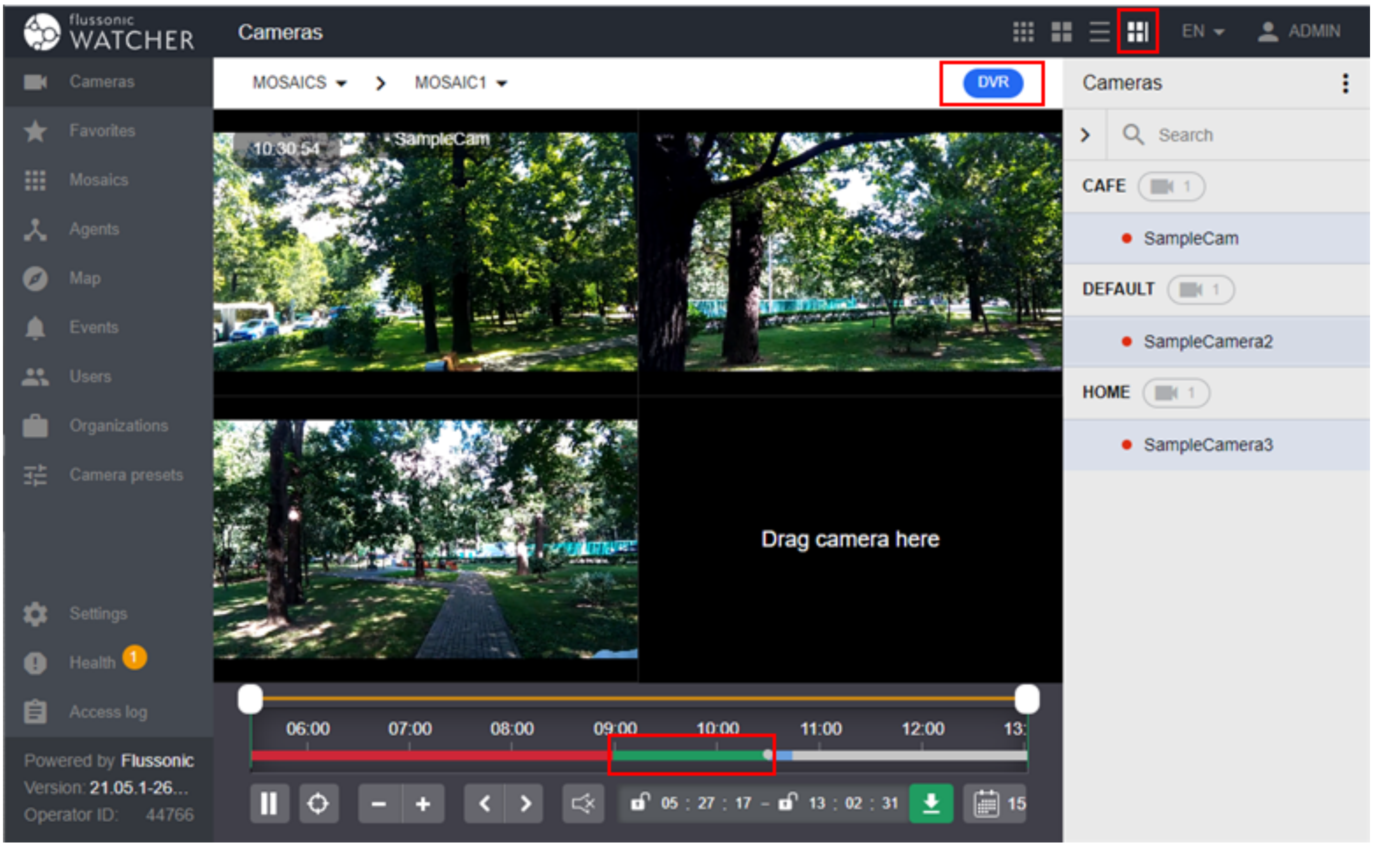


- Delete the camera



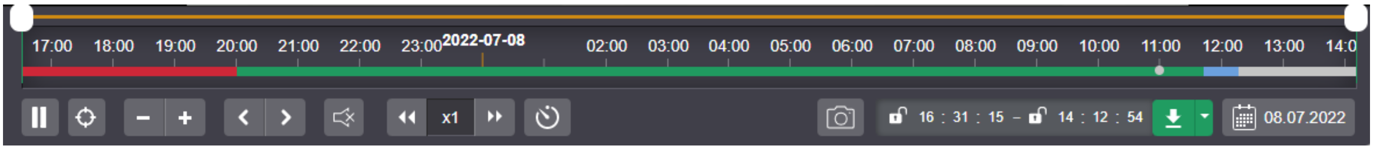
Video on the Dashboard

In **Dashbord** mode, live video is displayed when you switch to the mode. Click **DVR** to view archive from all cameras on the mosaic at once.



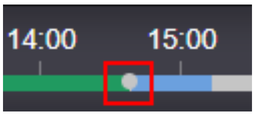
Playback controls

The playback control panel is in the lower part of the player box. The playback control panel may not display when DVR is disabled for the camera.



Controlling playback

When you open the player, the live video is played by default. The current playback position is shown as a grey dot on the timeline



.

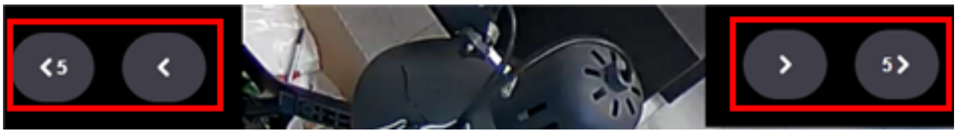
To pause the video (live or archived), click ⏸.

To resume playback or live view, click ▶.

Move the current playback position to the green area of the timeline (i.e. archive) to use the below playback controls.

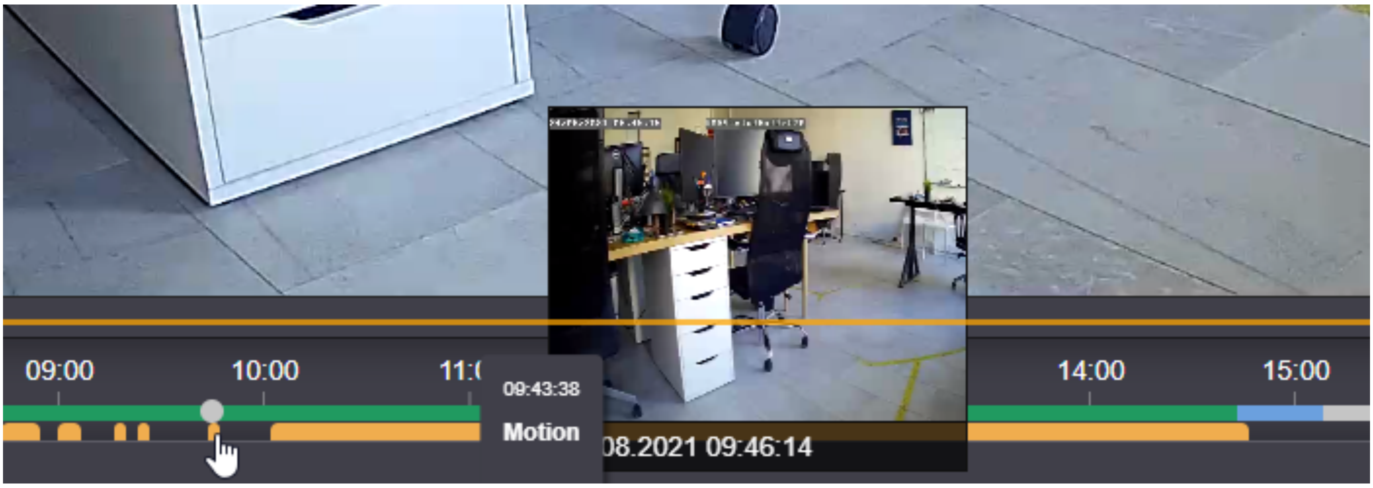
The ◀◀ x1 ▶▶ buttons allow for fast and slow playback of the archive. The current playback speed is displayed between the buttons. The minimum speed is x0.125, the maximum is x16.

The ⏱ button enables the frame-by-frame advance. When enabled, the playback automatically pauses and buttons appear on both sides of the frame for swiping frames forward and backward. You can view frames one by one or each fifth frame in both directions.



Browsing the timeline

The red parts of the timeline show periods without archive and the green ones are the periods where the archive is present. You may also find the marks of motion, LPR or face recognition events under the timeline. When you hover the mouse cursor over the green area, a screenshot of the moment is displayed next to the cursor with event explanation (if any).



Click any point in the green area to watch the archive starting from the selected moment. Click the blue or gray area to go back to live video.

Use the mouse wheel or the − + buttons to scale the timeline.

To move the timeline left or right, use the ‹ › buttons.

⌖ allows you to center the timeline on the current playback position.

It is convenient to use the calendar 📅 23.08.2021 for positioning in a rather large archive for several days or more. Days with archive are marked in light green. Click a day to start playing the earliest recording for that day.

Sound control

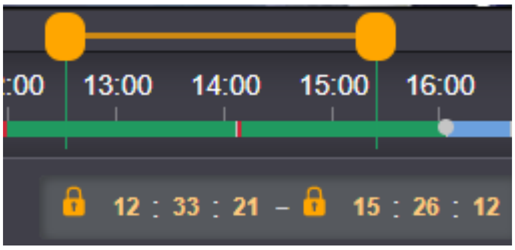
The sound control is a speaker icon. If the camera has a microphone, then the sound from it will be available in Watcher. Click this icon to enable/disable sound. When the icon looks like 🔇 it means that the sound is disabled. However, the sound is still recorded to the archive.

When the sound is enabled, you can control the volume: 🔈.

Fragment or frame export

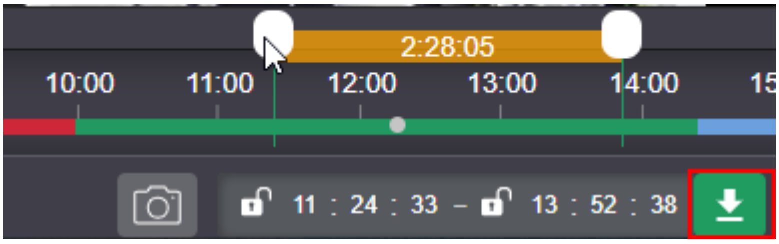
You can export the archive for the selected period to an MP4 file from the playback control panel. The period can be entered manually or set using the sliders above the timeline.

Click the lock icon next to the time in order to pin the slider: this is not necessary, but useful in case the timeline moves.



Hover the mouse over the slider or the line between sliders to see the fragment duration.

After choosing the period, click the download button.



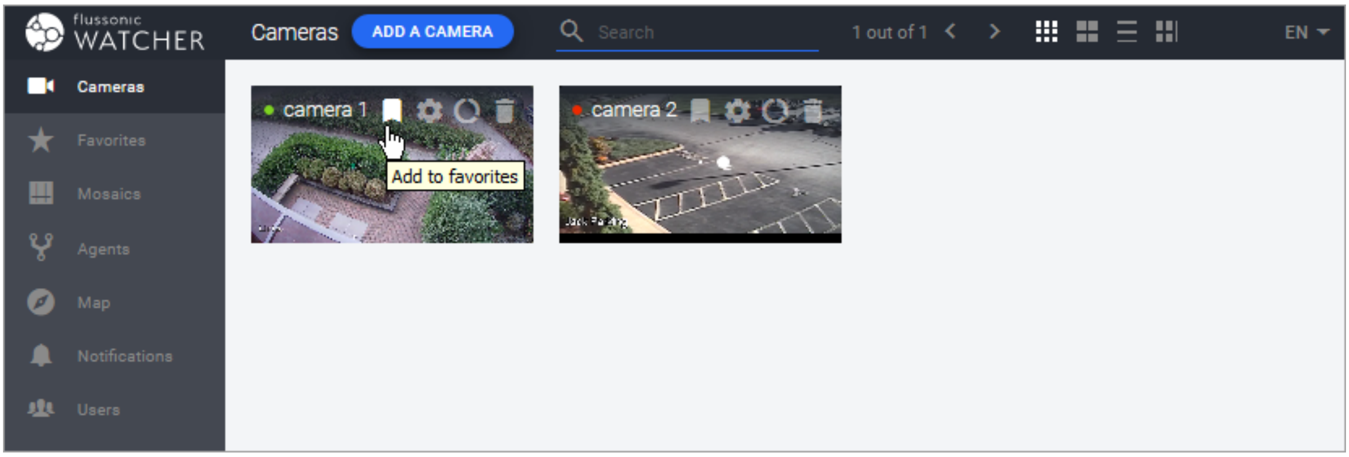
The button allows you to save the currently displayed frame of the archive in PNG format. The button is only displayed when current playback position is in the archive.

5.3 Adding a camera to Favorites

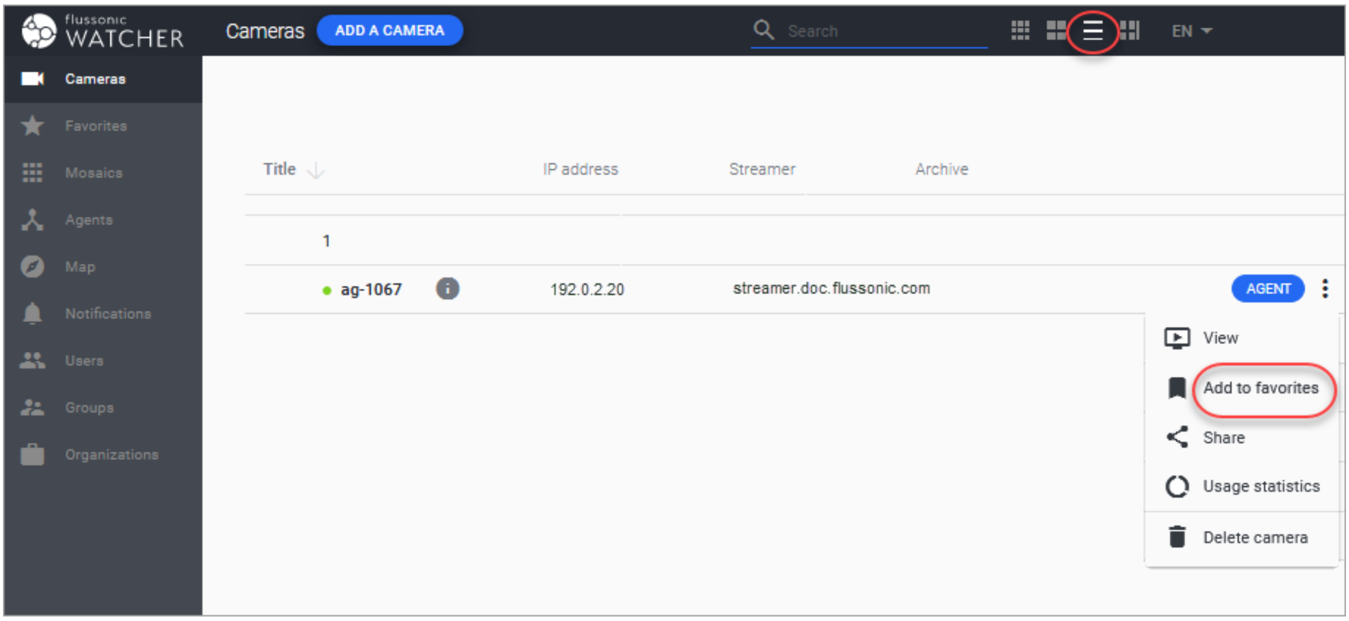
You can also add a camera to favorites to find it in one click.

To add a camera to the Favorites page:

1. Go to **Cameras**
2. In the list of cameras (which, by default, appears in Tiles mode), find the camera and click the bookmark icon on it:



Alternatively, in List mode, click the More button next to the camera and choose **Add to Favorites**.



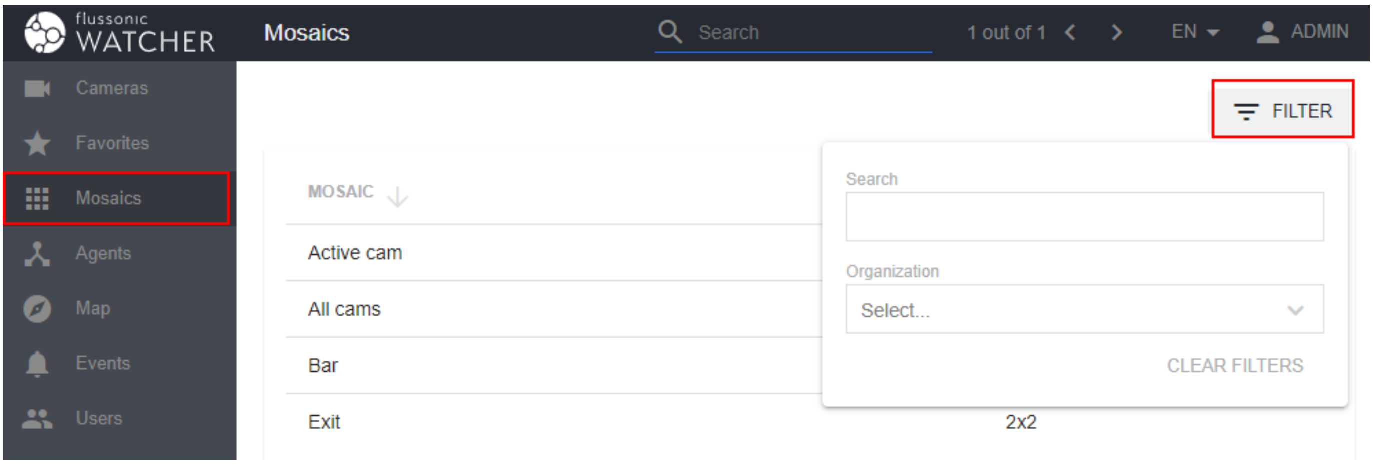
Title	IP address	Streamer	Archive	
1				
ag-1067	192.0.2.20	streamer.doc.flussonic.com		AGENT

The Favorites page is similar to the **Cameras** page with the same controls available.

5.4 Viewing mosaics

A **Mosaic** is a page with several simultaneously viewed cameras. Read here how to configure them.

Go to **Mosaics** menu item and select one of the previously created mosaic.

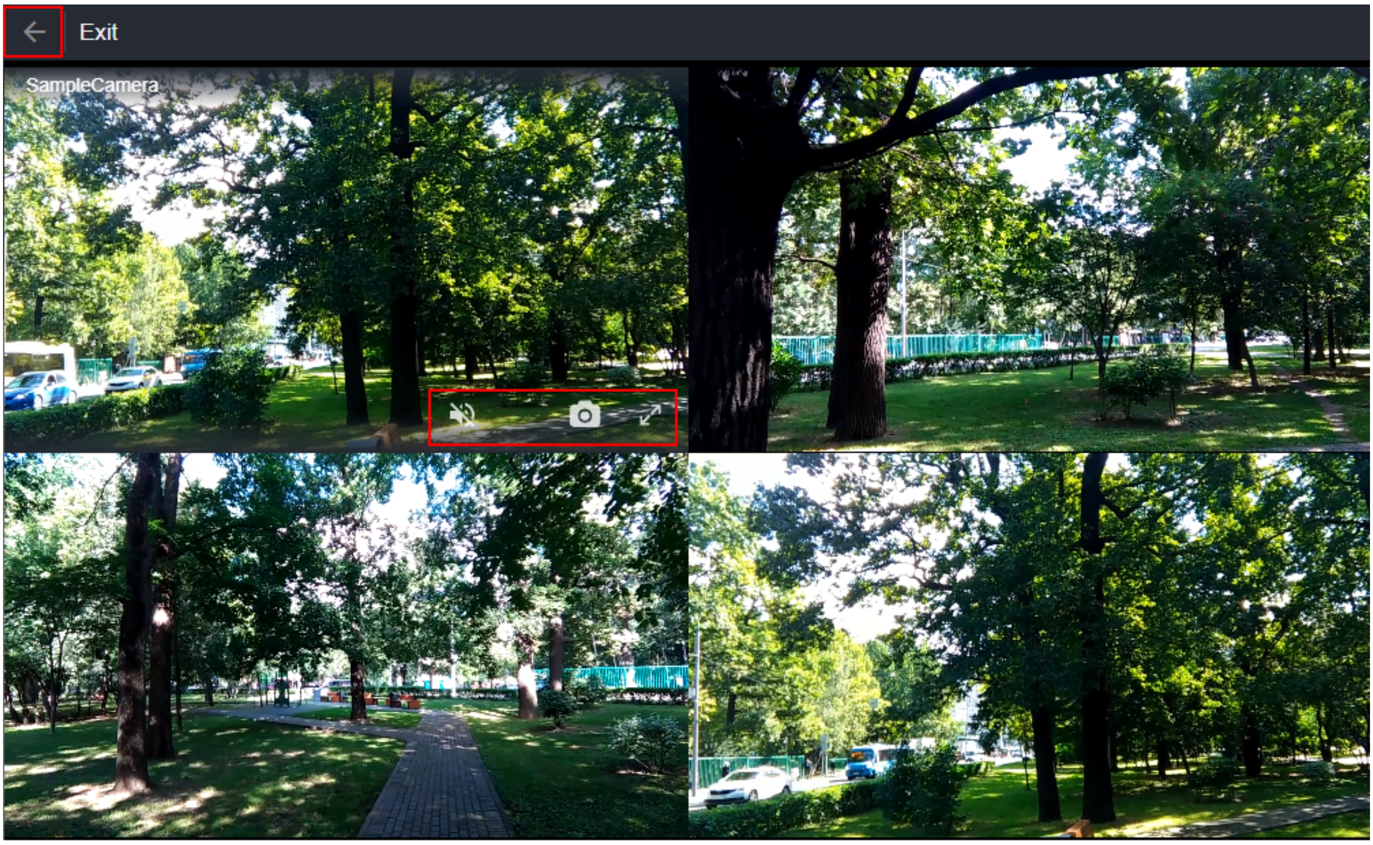


5.4.1 Mosaic UI

The mosaic interface hides the side menu to make it convenient to pin browser's tab on a dedicated monitor. To return to the list, click the arrow in the upper left corner.

The mosaic can use a sub-stream from the camera: the sub-stream will be shown in the grid, and when you click "expand" the player will open the stream in high quality.

Usuing sub-streams is the only way to play a large number of streams in real-time, as decoding video in full quality takes up a lot of CPU resources. There is no practical benefit either, as even a 4K monitor can only fit a 2x2 mosaic with cameras at a resolution of 1920x1080. In practice, you want to display at least 6-16 cameras on the monitor *(this means grids of 3x3 or 4x4)*, which gives about ~200-300px in height at best.

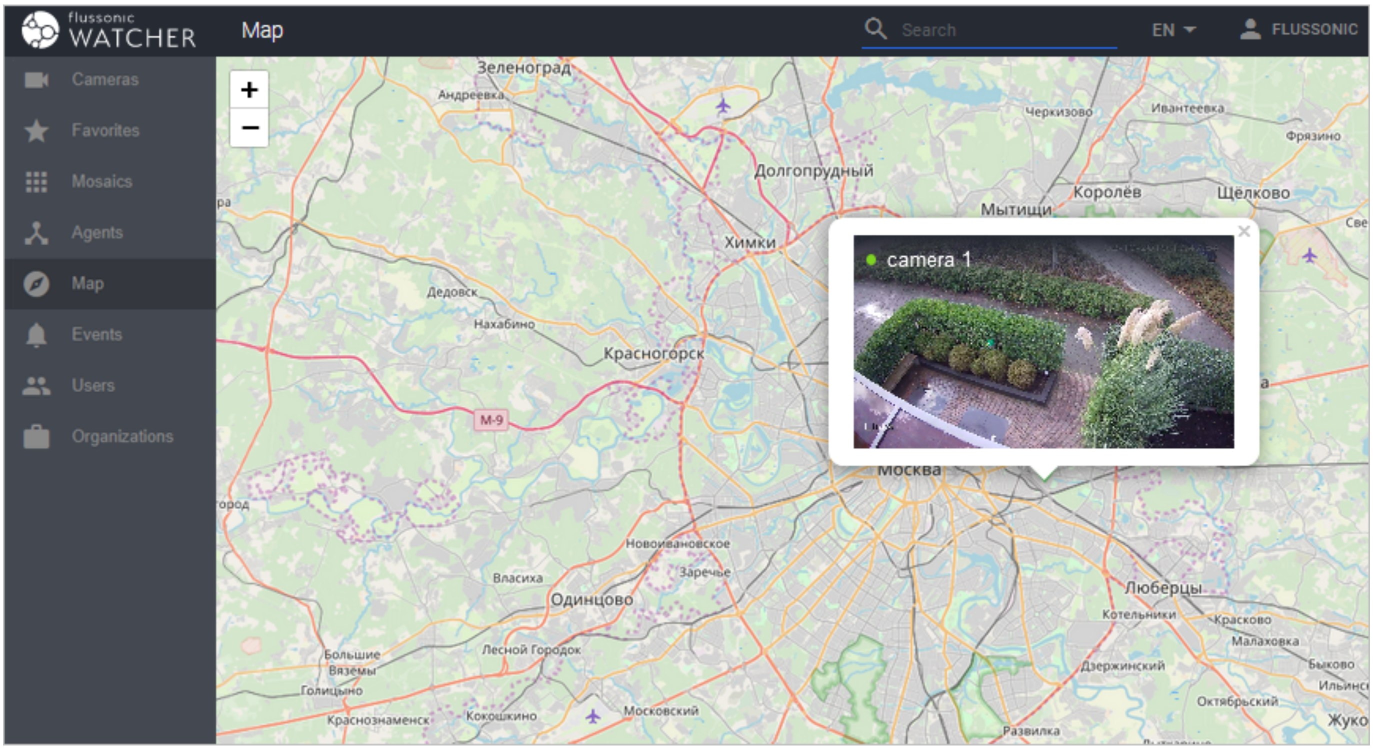


The buttons on the camera tile allow turning on/off the sound, save the frame to your computer and expand the video to full screen.

You can also use the Dashboard for advanced mosaic operation and control.

5.5 Map

The map on the page **Map** shows cameras that have geo coordinates configured in camera settings.



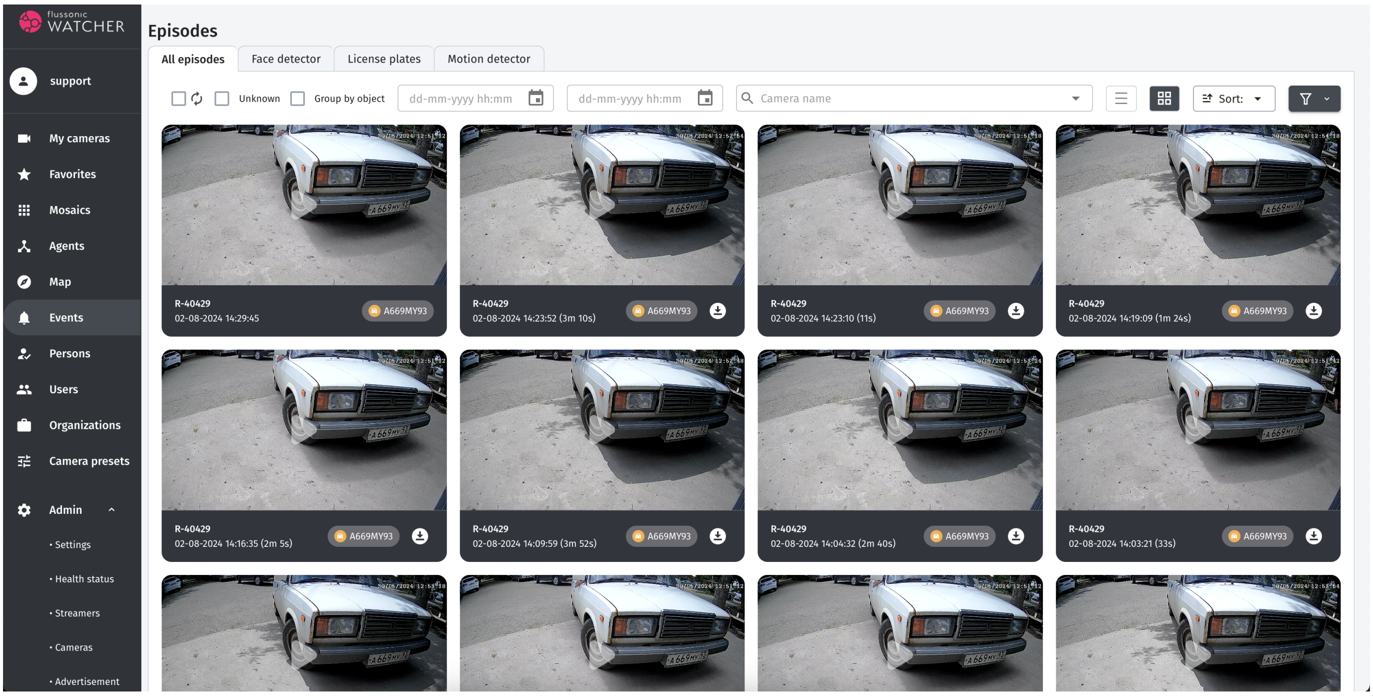
The map can also show floor plans if they are configured in the system.

The user must have permissions to cameras in order to see them on Map.

5.6 Events

The **Events** page shows the results of analytics subsystem operation. Events are grouped by three tabs:

- **Motion detector**: the tab shows motion detector triggering.
- **License plates**: the tab shows the list of recognized license plates.
- **Face detector**: the tab shows screenshots of recognized faces and provides addition tools to manage face lists and persons.



R-40429 02-08-2024 14:29:45 A669MY93

R-40429 02-08-2024 14:23:52 (3m 10s) A669MY93

R-40429 02-08-2024 14:23:10 (11s) A669MY93

R-40429 02-08-2024 14:19:09 (1m 24s) A669MY93

R-40429 02-08-2024 14:16:35 (2m 5s) A669MY93

R-40429 02-08-2024 14:09:59 (3m 52s) A669MY93

R-40429 02-08-2024 14:04:32 (2m 40s) A669MY93

R-40429 02-08-2024 14:03:21 (33s) A669MY93

> **Note**
>
> **Preview is not available** message over the screenshots means that archive for the period is already expired, and only database entry is left for the event.

You will find details about certain tabs in the articles on setting up the corresponding functions. General controls available for all types of events are described below.

You can use filters to search for events:

- **Camera**: search by full or partial camera name;
- **Search** by event name, vehicle license number or person's name, depending on the tab;
- **From** and **To**: search by time of an event registration being in a certain interval;
- **Organization**: search by Organization to which the camera belongs.

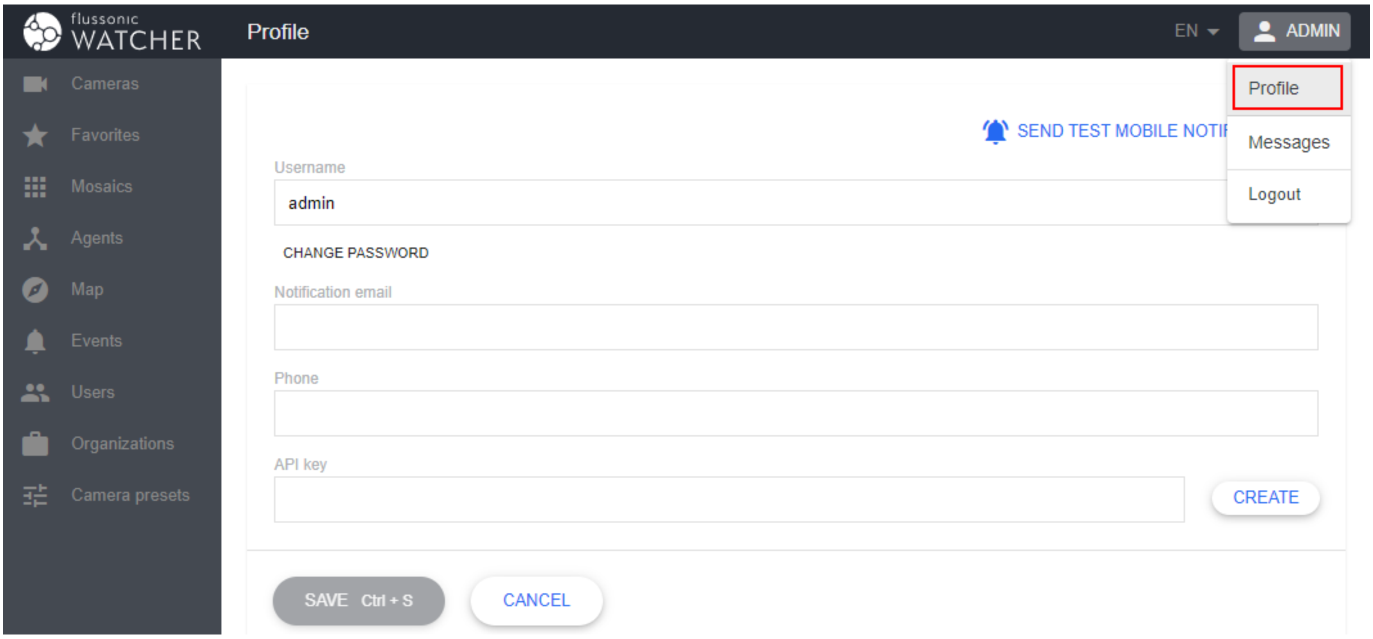
The list of events on the page is automatically updated while you set the criteria.

The **Auto update** checkbox is available on all tabs enabling automatic display of new events on the page. If the box is checked, new events are added to the top left of the list as soon as detected. If the box is cleared, you should refresh the page manually to view new events.

The **Motion detector** and **License plates** tabs provide the **Download** button to download the list of motion or license plate recognition events in .csv format.

5.7 Profile

Any Watcher user can edit their data for working with Watcher by selecting **Profile** in the top right corner of any Watcher web-UI page.



Username: admin

The following parameters can be changed:

- **Username**: User login
- Password (click **Change password**)
- **Notification email**: E-mail for notifications on password reset (the SMTP-server should be configured for that)
- **Phone** number
- **API key** to send API requests under specific user credentials.

This page also allows you to **send test mobile notification** to the mobile app (if you use it).

6. Developers Guides

6.1 API

6.1.1 Watcher API

You can integrate the Watcher video surveillance system into your system and create custom mobile applications using the flexible API.

Flussonic Watcher API let you import or export users and cameras individually or in bulk. API has the advanced integration options with client billing. It allows you manage the availability of cameras, private archives and billing, change passwords, check camera status and solve other tasks that are available in the GUI.

Also, the API let you configure authentication via RADIUS-server or use an authorization backend.

- Client APIv3
- Admin APIv3
- General description of APIv2
- API request authorization ways
- User Import API
- Camera Import API
- Integration with billing
- Change password
- Backend for user authorization
- RADIUS authorization

6.1.2 API request authorization

Getting the access token

Specify a JWT token in the HTTP Header in order for any Watcher API call to be executed. To get the token, use the corresponding request:

- For Watcher Client API: POST /watcher/client-api/v3/login
- For Watcher Admin API: POST /watcher/admin-api/v3/login

Example of the request to get the token:

```
curl -X POST -u user:password "http://localhost:80/watcher/client-api/v3/login"
```

In the response to this request, there are two parameters:

- `access_token` is a JWT token for executing API calls. It has a limited validity period encoded in the token itself.
- `refresh_token` is a long-lasting token that can be used to request a new `access_token`. When you implement your application, store the `refresh_token`, for example, in the database so that you can use it when needed.

Example of using `access_token`:

```
curl -X GET "http://localhost:80/watcher/client-api/v3/streams" \
    -H "Authorization: Bearer <access_token>" \
```

Refreshing the token without login and password

When the server returns `HTTP 401` to a request with the `access_token`, use `refresh_token` from the `/login` response to request the new token like that:

```
curl -X POST "http://localhost:80/watcher/client-api/v3/login" \
-H "Authorization: Bearer <refresh_token>" \
```

This way the user does not have to enter the login and password again.

Authorization in API v2

The old API v2 offered three types of authorization:

1. System **x-vsaas-api-key** from Watcher settings. It allows to execute any requests.

   `curl -v -X GET -H 'x-vsaas-api-key: 7c75da8fb314183f1f825271898a3687' http://127.0.0.1/vsaas/api/v2/cameras`

2. **x-vsaas-session** returned in response to `/vsaas/api/v2/auth/login` request in the **session** parameter. This key allows requests within the logged in user permissions.

   `curl -v -X GET -H 'x-vsaas-session: W98uOoiMFf46SyE78RjWIZjsaVM' http://127.0.0.1/vsaas/api/v2/auth/whoami`

3. User **x-vsaas-api-key** from user profile Use this key with **x-vsaas-user**: user name. It allows requests within the permissions of the specified user.

   `curl -v -X GET -H 'x-vsaas-api-key: M8rT4KvfT3tZpCj34Qbk5CEt' -H 'x-vsaas-user: user1' http://127.0.0.1/vsaas/api/v2/cameras`

6.1.3 User import API

You can import users and settings from third-party systems, databases or spreadsheets using API. It requires API authorization. To do that, include the APIKEY that can be found in the Watcher settings in the **x-vsaas-api-key** HTTP header.

To import a list of users, create a CSV file and send it to:

http(s)://YOUR_WATCHER_URL/vsaas/api/v2/users/import

CSV file fields:

- login — the main user ID for authentication
- email — used for password recovery.
- password — plain text password.
- is_active — 1: active user; 0: blocked user.
- is_admin — 1: administrator user; 0: regular user.
- note — a comment to this user.

Command-line import:

```
curl --data-binary @mydata.csv -H 'Content-type:text/plain; charset=utf-8' \
-H 'x-vsaas-api-key: YOUR_API_KEY' http://WATCHER-HOSTNAME/vsaas/api/v2/users/import
```

mydata.csv example:

```
login,email,password,is_active,is_admin,note
ivanov,ivanov@domain.tld,CergitMig,1,0,user1
petrov,petrov@domain.tld,LajQuolOy,0,1,user2
```

One string example:

```
echo -e "login,email,password,is_active,is_admin,note\nivanov,ivanov@domain.tld,CergitMig,1,0,user1\npetrov,petrov@domain.tld,LajQuolOy,0,1,user2" | curl
http://127.0.0.1:80/vsaas/api/v2/users/import?type=csv --data-binary @-  -H 'Content-type:text/plain; charset=utf-8' -H 'x-vsaas-api-key: 3a7d9386-6c3a-440d-
a75d-e6b3fdc3368e'
```

Response:

```
{"created":2,"updated":0,"deleted":0,"errors":{}}
```

6.1.4 Camera Import API

A POST request is used to import the cameras to the following URL: http(s)://YOUR_WATCHER_URL/vsaas/api/v2/cameras/import

Console:

curl 'http://localhost/watcher/client-api/v3/streams/import' \ -X POST \ --data-binary @mydata.csv \ -H 'Content-Type: text/csv' \ -H 'Authorization: Bearer eyJ0eXAiOiJKV1'

mydata.csv example:

name,title,comment,static,disabled,inputs_0_url,inputs_1_url,dvr_expiration,dvr_episodes_expiration,dvr_storage_limit,dvr_redundancy_factor,vision_alg,vision_areas,organization_id,coordinates_latitude,coordinates_longitude,is_test,onvif_ptz cam-e7fee73458,Primary camera title,,true,false,rtsp://127.0.0.1/stream/main,rtsp://127.0.0.1/stream/sub,86400,259200,32212254720,3,faces,x1 y1,1,55.71182944518981,37.7775426954031,false,false ,Secondary camera title,Camera comment,true,false,rtsp://127.0.0.1/stream2/main,rtsp://127.0.0.1/stream2/sub,172800,432000,,0,plates,x2 y2,1,51.711667039488155,35.77733281254769,true,true

Response:

```
{"created":2,"updated":0}
```

One string example:

echo -e "name,title,comment,static,disabled,inputs_0_url,inputs_1_url,dvr_expiration,dvr_episodes_expiration,dvr_storage_limit,dvr_redundancy_factor,vision_alg,vision_areas,organization_id,coordinates_latitude,coordinates_longitude,is_test,onvif_ptz cam-e7fee73458,Primary camera title,,true,false,rtsp://127.0.0.1/stream/main,rtsp://127.0.0.1/stream/sub,86400,259200,32212254720,3,faces,x1 y1,1,55.71182944518981,37.7775426954031,false,false ,Secondary camera title,Camera comment,true,false,rtsp://127.0.0.1/stream2/main,rtsp://127.0.0.1/stream2/sub,172800,432000,,0,plates,x2 y2,1,51.711667039488155,35.77733281254769,true,true" | curl 'http://localhost/watcher/client-api/v3/streams/import' -X POST --data-binary @mydata.csv -H 'Content-Type: text/csv' -H 'Authorization: Bearer eyJ0eXAiOiJKV'

Response:

```
{
{"created":2,"updated":0}
}
```

CSV or list of JSON objects with the following fields:

- title: camera name.
- name: stream name. Default is ‘title’ with a random suffix.
- static: 1: constantly working stream, 0: on-demand stream.
- stream_url: main stream RTSP URL.
- substream_url: secondary stream RTSP URL (for multi-bitrate). Turned off by default.
- thumbnails (string): camera snapshot URL. Send 0 instead of the URL to reset thumbnails. If you don’t know the camera snapshot URL, use 1. Watcher will turn it on automatically. Note: we recommend that you use a direct URL. Otherwise, the server load will increase at 10%. If you specify the URL, a direct communication with the camera is established, saving your server processing power.
- onvif_url: camera URL to access via the onvif protocol. By default, it’s set to ‘no.’
- onvif_profile: ONFIV profile
- ptz: (0 or 1) — turn PTZ off/on (if camera supports).
- access: camera access type (private / public / authorized). Public will be accessible to all users, private — to a camera owner only. Private is a default value.
- owner: camera owner login.
- enabled: (0 or 1) turns camera on / off.
- dvr_path: path to save the archive. Default is no archive.
- dvr_depth: (integer in days) — number of days to store the archive. 0 — disables the archive.
- coordinates:geographical coordinates.
- postal_address: camera’s postal address.
- comment: camera comment.
- agent_model (string): camera model.
- agent_serial (string): camera serial number.
- agent_id (string): camera agent unique number.
- agent_key (string): special field used for Watcher authorization.
- agent_pin (string): special field used for Watcher authorization.

6.1.5 Integration with existing billing system

This article describes common scenarios of integration between Flussonic Watcher and third party system allowing managed camera sales and accounting for subscribers and their services. API for integration is fully available here.

Explanation of terms:

- **Provider** is a client of Flussonic, the owner of the service
- **Subscriber** is a Provider's subscriber, the user of the service
- **Billing** is a third party system not related to Watcher; it allows managing plans (billing plans, subscriptions) of Provider's services to subscribers as well as payment collection

The concept of billing implies that the billing service is the master system storing and managing subscribers data, and not Flussonic Watcher. Such a best practice allows you to centrally manage services in different systems, linking, for example, smart home and video surveillance in a single project.

Before you add any cameras for you subscribers, make sure to create the corresponding structure of organizations and presets in Watcher, assign permissions to subscriber. You can do it in Watcher web UI and/or by corresponding API requests for organizations, users, folders management. The mechanism for matching billing plans and Watcher presets, as well as the logic that will affect access, archive depth and other settings, is on the billing side in any case.

The integration scenario is different for cameras with or without the Agent. Both are described below.

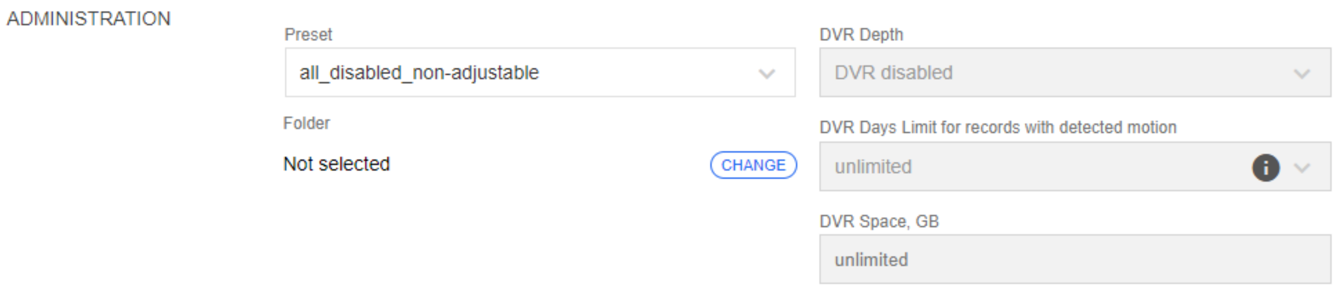
You will also find below example requests to suspend Watcher services for a subscriber.

Creating presets

A preset is a predefined set of archive and analytics settings to populate to the camera. Presets correspond to your billing plans, so you should create and configure presets before adding cameras.

The subscribers usually shall not be allowed to change the camera settings populated from the preset. Just make the preset non-configurable to fulfill this requirement.

With non-configurable presets, the subscribers will not be able to change the preset-defined parameters even if they have the Organization Owner permissions allowing them to edit the camera settings. The settings will be visible but not editable.



ADMINISTRATION

Preset: all_disabled_non-adjustable

Folder: Not selected

DVR Depth: DVR disabled

DVR Days Limit for records with detected motion: unlimited

DVR Space, GB: unlimited

Organization Owners *cannot* create, edit or delete presets: only Watcher Administrators have such permissions. However, if several presets are linked to an Organization, then the Organization Owner will be able to switch presets in the camera settings.

Use the following API request to create a preset from the billing system:

POST http(s)://YOUR_WATCHER_URL/vsaas/api/v2/presets

`title` is a required parameter for the preset name.

Request example to create a non-adjustable preset (is_adjustable=false) defining the archive depth of 2 days:

```
curl -v POST http://127.0.0.1/vsaas/api/v2/presets \
-d '{"title":"non-adjustable preset","is_adjustable":false, "organization_id":"1", "dvr_depth": 2}' \
-H 'x-vsaas-api-key: 7c75da8fb314183f1f825271898a368d' \
-H "Content-Type: application/json"
```

The response will contain all the parameters of the created preset, including `id` for further use in the camera management requests.

```
{
    "is_default": false,
    "is_deleted": false,
    "title": "non-adjustable preset",
    "dvr_space": null,
    "id": 1,
    "is_adjustable": false,
    "dvr_depth": 2,
    "vision_areas": null,
    "precise_thumbnails_days": 0,
    "vision_enabled": null,
    "vision_alg": null,
    "vision_gpu": null,
    "domain_id": 1,
    "dvr_lock_days": null,
    "vision_params": {}
}
```

You can also use requests like **GET/PUT/DELETE http(s)://YOUR_WATCHER_URL/vsaas/api/v2/presets/{preset_id}** to get/change/delete the preset with the specified ID.

Cameras with Agent

When you use cameras with Agent, the scenario is as follows:

1. A modified firmware with Flussonic Agent is uploaded on a batch of cameras (Flussonic specialists can do that). This firmware contains information about which Flussonic Watcher this camera should be linked to.
2. The Provider enters camera info into the **billing** inventarization system while the camera is on stock.
3. When selling the camera to the subscriber, the Provider's employee links the camera serial number in **billing** with the subscriber's ID.
4. As soon as the camera connects to some network, it receives authorization data from the activation server. This data is not related to the subscriber's authorization; this is the camera authorization.
5. The activated camera immediately starts trying to connect to Flussonic Watcher.
6. The activation server sends data about the camera to the **billing**.

> **Note**
>
> The activation server can send data about the camera directly to Watcher, if information about its owner is already known when activating the camera with the Agent (for example, when IRIS camera is activated from the mobile app). It depends on the Agent type and on requirements to integration. Please contact Flussonic Watcher technical support to configure the activation server.

7. **Billing receives** information about the newly created camera, adds information about the preset (billing plan) and the organization to it.
8. **Billing sends** information about newly created camera to Watcher. This and the previous steps are to be implemented at billing side.
9. Now the camera can connect to Watcher and start streaming.

The subscriber does not have to change any settings of routers, cameras and other network devices when the process is arranged the above way. The camera appears in the subscriber's account automatically as soon as it is connected to the network as soon as the API is implemented in billing for receiving data about newly created cameras and sending this data to Watcher. Such a data proxying scheme is needed to add information about the owner of the camera and the services that are available on this camera.

The activation server maintained by Flussonic sends a request with a CSV or a list of JSON objects to the configured URL. Please contact Flussonic Watcher technical support to make the server send requests to the billing URL.

Send all data received from the activation server to Flussonic Watcher unchanged unless you decide to change them for some reason. For example, you may receive the can_ptz=1 flag and set it to 0 to forbid the subscriber from controlling PTZ in Watcher.

The fields (in CSV or JSON) sent from the activation server to the billing:

- **agent_model** (string): camera model
- **agent_serial** (string): camera serial number
- **agent_id** (string): unique ID of the Agent on camera
- **agent_key** (string): special field for the camera authorization by Watcher
- **stream_url** (string): main RTSP URL of the stream
- **substream_url** (string): second RTSP URL of the stream
- **thumbnails** (string): thumbnails URL
- **onvif_url** (string): URL for camera response via ONVIF
- **onvif_profile** (string): service field
- **can_ptz** (0 or 1): enable or disable PTZ
- **abonent_sign** (integer): encrypted information about the organization and the user who owns the camera. It is present only for the corresponding type of Agent, when the subscriber manually adds a camera to the organization via Watcher mobile app or web UI. If this field is filled, you do not have to add info about the camera owner.

> ⚠ **Warning**
>
> Having received the data, the billing system must return a "200" response to the activation server. Otherwise, the activation server will retry sending data until this acknowledgment is received.

The information received from the activation server should be "merged" with the camera data already existing in the billing or other accounting system by parameter **agent_serial** (camera serial number). You should understand that **agent_id** can change, e.g. if the camera is handed to another subscriber. The serial number of the camera does not change.

> ✎ **Note**
>
> Thus, if there is an inventory system in billing, in which the camera is linked to a subscriber before the first activation, then a new record will not appear; instead, you must fill in the missing fields in the existing line in the database. If there is no such inventory system, then you need to create new entries for cameras in the billing when you receive an activation message.

The billing should add additional camera attributes, such as linking to an organization or details on services within the plan.

When the additional fields (**preset_id**, **enabled**, **organization_id**) are added, send the information about the camera (CSV or JSON) to Watcher using one of the following requests:

POST http(s)://YOUR_WATCHER_URL/vsaas/api/v2/cameras/import to add several cameras

POST http(s)://YOUR_WATCHER_URL/vsaas/api/v2/cameras to add one camera

Example request for several cameras is shown below. Please note that the **X-Vsaas-Api-Key** should be changed to the one shown in your Watcher settings.

```
curl -v POST http://localhost:80/vsaas/api/v2/cameras/import \
  -d \
  '[{
  "name":"cam1","stream_url":"fake://clock",
  "enabled":true,"dvr_depth":3,"agent_id":"123098456","agent_serial":"mJ0ODnktZFc",
  "agent_key":"salt:secretkey","preset_id":"1","organization_id":"1"
  }]' \
  -H "X-Vsaas-Api-Key: 7ab056b1-5bb1-4501-b528-d69538392842" \
  -H "Content-Type: application/json"
....
{
  "deleted": 0,
  "updated": 0,
  "errors": {},
  "created": 1
}
```

If you do not specify the preset ID and organization ID in the request, the camera will be added to the default organization with the default preset.

Cameras without Agent

The scenario for cameras without Agent (e.g. RTSP, ONVIF) would usually be as follows:

1. A camera is connected to the internal Provider's network.
2. The subscriber asks Provider to grant access to the camera.
3. The Provider sends request to **billing** to add the camera for the subscriber or give permissions to use the camera.
4. If it is a new personal camera (e.g. for subscriber's smart home):
    - It is assumed that the user and organization for the subscriber have already been created in Watcher. If they are not, you or the billing system need to create them in the Watcher web UI or by sending the appropriate API requests from billing.
    - **Billing** fills in the necessary camera attributes in accordance with the billing plan.
    - **Billing sends** a request to Watcher to add a camera to the subscriber (see example below).
5. If the subscriber wants to connect to a shared camera, for example, to an intercom or to the Safe City system:
    - It is assumed that the corresponding camera and organization have already been created in Watcher. If they are not, you or the billing system need to create them in the Watcher web UI or by sending the appropriate API requests from billing.
    - **Billing** fills in the necessary attributes of the user (subscriber) by adding to the organization to which the required camera is added.
    - **Billing sends** a request to Watcher to create or update the subscriber (see example below).

ADDING A CAMERA

POST http(s)://YOUR_WATCHER_URL/vsaas/api/v2/cameras/import to add several cameras.

POST http(s)://YOUR_WATCHER_URL/vsaas/api/v2/cameras to add one camera.

The following parameters are required in the request to add a camera:

- **preset_id** (integer): Watcher preset identifier corresponding to the billing plan in billing.
- **stream_url** (string): main RTSP URL of the stream.
- **organization_id** (integer): organization identifier. The organization must have already been created in Watcher.

The simple example of a request to add one camera is shown below. You can try to copy this request and execute it on a server with Watcher installed. Please note that the **X-Vsaas-Api-Key** should be changed to the one shown in your Watcher settings.

```
curl -v -X POST http://127.0.0.1/vsaas/api/v2/cameras \
-H 'x-vsaas-api-key: 7c75da8fb314183f1f825271898a368d' \
-H 'content-type: application/json' \
-d '{
  "title": "myCamera",
  "stream_url": "fake://clock",
  "preset_id": "1",
  "organization_id": "1"}'
```

As a result of such a request, a virtual camera called "myCamera" broadcasting time ("stream_url": "fake://clock") will be added to the default organization ("organization_id": "1") with default preset ("preset_id": "1"). In response to the request, Watcher will send JSON with a list of all parameters of the added camera.

ADDING A USER

POST http(s)://YOUR_WATCHER_URL/vsaas/api/v2/users

Example:

Please note that the **X-Vsaas-Api-Key** should be changed to the one shown in your Watcher settings.

```
curl -v POST http://127.0.0.1/vsaas/api/v2/users \
-d '{"login":"user1", "organization_id":"1"}' \
-H 'x-vsaas-api-key: 7c75da8fb314183f1f825271898a368d' \
-H "Content-Type: application/json"
```

As a result of this query, the user named user1 will be added to the default organization ("organization_id":"1").

The unencrypted password can also be sent in this query (use "password" parameter). However, if the Provider does not store subscriber passwords as plain text and cannot transfer them in any way, you need to configure external authorization backend in order for the subscriber to be able to log into Watcher.

See below for an example of updating the user.

Disabling the camera to the subscriber from billing

If the subscriber, for some reason, should no longer use the camera (for example, turned off the service or did not pay on time), billing should send a corresponding request to Watcher.

The easiest way is disabling the user. You can use the following request **PUT https://watcher_api_url/vsaas/api/v2/users/(int:user_id)** with "enabled": false.

Example:

```
curl -v -X PUT http://127.0.0.1/vsaas/api/v2/users/5 \
-H 'x-vsaas-api-key: 7c75da8fb314183f1f825271898a3687' \
-H 'content-type: application/json' -d '{"enabled":false}'
```

There are also the following main options for limiting rights without disabling the user:

1. The subscriber's personal camera, and the subscriber has no rights to edit cameras.

   Disable the camera by **PUT https://watcher_api_url/vsaas/api/v2/cameras/(string:cam_name)**, with "enabled": false.

   Example:

   ```
   curl -v -X PUT -H 'x-vsaas-api-key: 7c75da8fb314183f1f825271898a3687' -H 'content-type: application/json' \
   http://127.0.0.1/vsaas/api/v2/cameras/inst1cam-8e1a6e020e -d '{"enabled":false}'
   ```

   > **Note**
   >
   > Please note that the name of the camera in the interface may not coincide with its name in the database.

2. The subscriber's personal camera, and the subscriber has rights to edit cameras.

   Prevent the user from editing cameras by request **PUT https://watcher_api_url/vsaas/api/v2/users/(int:user_id)**, with "readonly": true.

   ```
   curl -v -X PUT -H 'x-vsaas-api-key: 7c75da8fb314183f1f825271898a3687' -H 'content-type: application/json' \
   http://127.0.0.1/vsaas/api/v2/users/5 -d '{"readonly": true}'
   ```

   Then disable cameras as shown above in option 1.

3. The camera is publicly available, many subscribers use it (for example, a video intercom in an apartment building or a camera of the "Safe City" system)

   Please note that you cannot restrict access to a particular camera for a user, you can only block access to a folder. See this page for details.

   The request to delete permissions to access a folder is **DELETE https://watcher_api_url/vsaas/api/v2/organizations/(int:organization_id)/folders/(int:folder_id)/users/(int:user_id)**

   Example request to restrict access for user with ID=3 to the folder with ID=2 in the organization with ID=1:

   ```
   curl -v -X DELETE  -H 'x-vsaas-api-key: 7c75da8fb314183f1f825271898a368d' -H 'content-type: application/json' \
   http://127.0.0.1:80/vsaas/api/v2/organizations/1/folders/2/users/3
   ```

6.1.6 Backend for user authorization

Users (subscribers) must be pre-configured according to the Watcher structure

How it works:

- Provider implements the HTTP request handler that has the logic to authenticate subscribers.
- Provider enters the path to the Flussonic Watcher authorization backend (Settings — External authentication).
- A subscriber logs into Flussonic Watcher using login/password as per backend database.
- Watcher transfers this data to the backend in a request payload.
- Backend checks the incoming data to make a decision on the subscriber authorization:
    - If the subscriber is authorized, backend returns the 200 HTTP code.
    - If the authentication data is incorrect, backend returns the 403 HTTP code.
    - If the subscriber is not found, the system returns 404.
    - If the authentication backend was out of reach or was not able to respond within 2 seconds, the subscriber's credentials are verified by the Watcher’s database.

> **Note**
>
> If the subscriber's password in the backend database is different from Flussonic database, then the password cache in the Flussonic database is replaced with the backend password cache upon successful authorization via backend. Thus, the subscriber will be able to access the system with a single password even if backend is unavailable.
>
> However, this leads to the fact that subscribers will not be able to change their passwords via Watcher; even if they try, the new password will not work and will be overwritten at next successful authorization via backend.

```
import falcon, json

class AuthResource:
  def on_get(self, req, resp):
    print "GET %r\n%r" % (req.uri, req.params)
    login = req.params.get('login', None)
    password = req.params.get('password', None)
    if not login or not password:
      print 'incorrect request login: %r, pass: %r' % (login, password)
      resp.status = falcon.HTTP_400
      return

    if login == 'user0':
      if password == 'letmein':
        return
      resp.status = falcon.HTTP_403
      return

    if login == 'user1':
      if password == 'letmein':
        return
      resp.status = falcon.HTTP_403
      return

    resp.status = falcon.HTTP_404

app = falcon.API()
ad = AuthResource()

app.add_route('/auth', ad)
```

Examples

A user can get through:

```
curl -vvv http://localhost:8001/auth\?login\=user0\&password\=letmein

*   Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 8001 (#0)
> GET /auth?login=user0&password=letmein HTTP/1.1
> Host: localhost:8001
> User-Agent: curl/7.47.0
> Accept: */*
>
< HTTP/1.1 200 OK
< Server: gunicorn/19.7.0
< Date: Mon, 20 Mar 2017 10:16:21 GMT
< Connection: close
< content-length: 0
< content-type: application/json; charset=UTF-8
<

* Closing connection 0
```

A user can’t get through:

```
curl -vvv http://localhost:8001/auth\?login\=user0\&password\=wrong

*   Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 8001 (#0)
> GET /auth?login=user0&password=wrong HTTP/1.1
> Host: localhost:8001
> User-Agent: curl/7.47.0
> Accept: */*
>
< HTTP/1.1 403 Forbidden
< Server: gunicorn/19.7.0
< Date: Mon, 20 Mar 2017 10:16:27 GMT
< Connection: close
< content-length: 0
< content-type: application/json; charset=UTF-8
<

* Closing connection 0
```

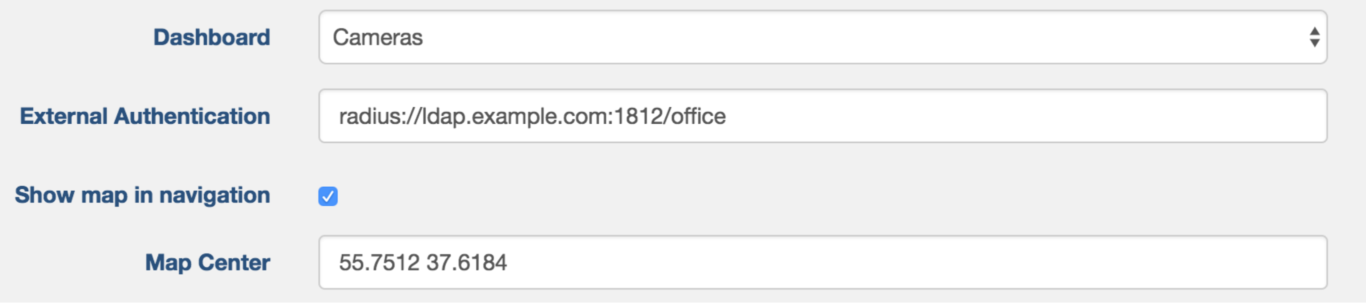
A user is not found:

```
curl -vvv http://localhost:8001/auth\?login\=user10\&password\=wrong

*   Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 8001 (#0)
> GET /auth?login=user10&password=wrong HTTP/1.1
> Host: localhost:8001
> User-Agent: curl/7.47.0
> Accept: */*
>
< HTTP/1.1 404 Not Found
< Server: gunicorn/19.7.0
< Date: Mon, 20 Mar 2017 10:20:04 GMT
< Connection: close
< content-length: 0
< content-type: application/json; charset=UTF-8
<

* Closing connection 0
```

6.1.7 RADIUS authentication

RADIUS server can be used to authenticate Watcher users. It is especially useful, if you have a large number of users.

The setting can be enabled via admin interface:



Dashboard	Cameras
External Authentication	radius://ldap.example.com:1812/office
Show map in navigation	☑
Map Center	55.7512 37.6184

The address 'radius://ldap.example.com:1812/secret' consists of 3 parts: host, port and secret. Change it according to your RADIUS server settings. Now, when a user try to login, Watcher redirects to the server via RADIUS protocol. Watcher sends **User-Name** and **User-Password** in the Access-Request query.

- Watcher redirects to RADIUS on every user log in.
- If the RADIUS answers Access-Accept, Watcher logs user in and saves the HEX password to the database.
- If the RADIUS answers Access-Reject, the user becomes locked in the database.
- If the RADIUS did not answer, Watcher searches a user in the database.

It is necessary to bear in mind that RADIUS should know about all users, including administrators. The administrator user attribute can not be transferred to the RADIUS response and it can be assigned through Watcher only.

6.1.8 Simple Event collector

The application accepts the POST-request, writes the JSON-data in a file */tmp/event.txt* and sends JSON back to Watcher.

```
from flask import Flask, request, jsonify
import json

app=Flask(__name__)

@app.route('/events',methods = ['POST'])
def events():
    d = request.get_json(force=True)

    with open("/tmp/events.txt", "a") as write_events:
        write_events.write(str(d)+'\n')

    return jsonify(d)

app.run(host = '0.0.0.0',debug=True)
```

gunicorn is required for launching the application, install it using following command: *pip3 install gunicorn*

Run command, for example:

```
gunicorn --bind 0.0.0.0:5000 wsgi:app -D -w 3 --log-syslog --reload -g www-data -u www-data --log-file /var/log/event_coll.log --error-logfile /var/log/
event_coll_error.log
```

6.1.9 Integrating Watcher with access control systems

Flussonic Watcher can send commands to access control systems (ACS) so that doors can be open after recognizing a person from the list. Below you can find a script for integration with the Sigur access control system, with comments on its use.

```
import socket
import sys
import argparse
import http.server
import socketserver
import cgi
import json
import requests
import logging
import logging.config
from logging.handlers import TimedRotatingFileHandler

fh = TimedRotatingFileHandler("ACS_integration.log", when='midnight')
sh = logging.StreamHandler()
logging.basicConfig(handlers=(fh, sh),
                    format='[%(asctime)s.%(msecs)03d | %(levelname)s]: %(message)s',
                    datefmt='%d.%m.%Y %H:%M:%S',
                    level=logging.INFO)

class ACS_Sigur:
    @staticmethod
    def connect(ip, port):
        sigur = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        try:
            sigur.connect((ip, int(port)))
        except:
            print("Connection error: ", sys.exc_info())
        else:
            print("Sigur server connected!")
        return sigur

    @staticmethod
    def login(sigur):
        message = "LOGIN 1.8 Administrator\r\n" #Change your credentials to connect to Sigur
        sigur.send(bytes(message, 'utf-8'))
        reply = sigur.recv(1024)
        data_reply = reply.decode('ascii')
        data_reply.replace('\n','')
        if "OK" in data_reply:
            print("Login in server successfull")
        else:
            print("Sigur server is not connected. That is why:", data_reply)

    @staticmethod
    def open_door(sigur):
        message = "ALLOWPASS 1 2 IN\r\n" #Change ID of a door you want to open. You can find ID in a Sigur app.
        sigur.send(bytes(message, 'utf-8'))
        reply = sigur.recv(1024)
        data_reply = reply.decode('ascii')
        data_reply.replace('\n','')
        if "OK" in data_reply:
            print("Door is opened")
        else:
            print("Something went wrong. That is why:", data_reply)

def create_cmd_parser():
    parser = argparse.ArgumentParser()
    parser.add_argument('-ACS', action="store", dest="ACS")
    parser.add_argument('-ip', action="store", dest="ip")
    parser.add_argument('-port', action="store", dest="port")
    parser.add_argument('-serverport', action="store", dest="serverport")

    return parser

parser = create_cmd_parser()
args = parser.parse_args()

class MyHandler(http.server.BaseHTTPRequestHandler):
    def do_POST(self):
        logging.info("New request from {client}".format(client = self.client_address))
        content_length = self.headers.get('content-length')
        if content_length == None:
            result = ""
        else:
            body = self.rfile.read(int(content_length))
            result = json.loads(body, encoding='utf-8')
        logging.info("Request JSON is: {result}".format(result = result))
        if args.ACS == "Sigur":
            logging.info("Trying connect to the Sigur server")
            try:
                sigur = ACS_Sigur.connect(args.ip, args.port)
                logging.info("Sigur server connected")
                ACS_Sigur.login(sigur)
                logging.info("Success login on a Siger server")
                try:
                    ACS_Sigur.open_door(sigur)
                    logging.info("Success login on a Sigur server")
                except:
                    logging.warning("There was a problem while opening door")
                sigur.close()
                logging.info("Connection with the Sigur server closed")
            except:
                logging.warning("Something went wrong. Basic request is nor JSON: {body}".format(body = body))

        if args.ACS == "Beward":
            logging.info("Trying connect to the Beward")
            uri = 'http://'+args.ip+'/cgi-bin/io/port.cgi?action=O0:/'
            try:
                response = requests.get(uri, auth=('admin', 'admin'))
                if response.status_code == 200:
                    logging.info("Door was opened")
                else:
                    logging.warning("Something went wrong")
            except:
                logging.warning("There was an error while sending command to Beward")

def main():
    logging.info("Application started ACS - {ACS}, IP address - {ip}, port - {port}, serverport - {serverport}".format(ACS = args.ACS, ip = args.ip, port =
args.port, serverport = args.serverport))
    if args.ACS == "Sigur":
        logging.info("ACS Sigur chosen")
    elif args.ACS == "Beward":
        logging.info("ACS Beward chosen")
    else:
        logging.info("ACS is unknown. Please use \"Sigur\" or \"Beward\".\nIntegrated module shutted down.")
        sys.exit()
    try:
        with socketserver.TCPServer(("127.0.0.1", int(args.serverport)), MyHandler) as httpd:
            logging.info("Started listening port {port}".format(port = args.serverport))
            httpd.serve_forever()
    except:
        logging.warning("Cannot connect to the listening port or someone shut application down")
        sys.exit()

if __name__ == "__main__":
    main()
```

How to use the script for access control integration:

1) Change a script listing according to your Sigur installation. You'll need to set Sigur login and password by editing the line:

```
message = "LOGIN 1.8 Administrator\r\n"
```

2) Set the ID of the door you want to open:

```
message = "ALLOWPASS 1 2 IN\r\n"
```

3) Change a systemd configuration file for your service to provide autostart:

```
[Unit]
Description=ACS Unlocker
After=network.target

[Service]
Type=simple
Restart=always
RestartSec=3
User=root
Group=root
WorkingDirectory=/opt/acs-unlocker
ExecStart=/opt/flussonic/bin/python3 runner.py -ACS Sigur -ip {IP} -port 3312 -serverport {serverport}

[Install]
WantedBy=multi-user.target
```

You'll need to set the `{IP}` of a Sigur server and the `{serverport}` on the server with installed integration module that will listen to events from Flussonic Watcher Face Recognition module and send it to Sigur.

Note: You will need to specify the `{serverport}` port when configuring events notification subscription in a face recognition module.

4) Reload and restart your new systemd service.

5) Subscribe to face recognition events by using Flussonic Watcher API, which sends events on the server port that you chose in a `.service` file. The face recognition events have `person_detected` type, but if you are implementing your access control system based on LPR then use `car_detected`.

Example of the request to create the subscription to the notifications about the person with `id=1`:

```
curl --header "Content-Type: application/json" \
--request POST \
-d '{"camera_id": "person.detection.test.camera-7d8ea4ebf2",
"notification_type": "http",
"event_type": "person_detected",
"webhook_params": {
  "url": "http://example.com",
  "method": "post",
  "params": {
      "mode": "single",
      "id": 1
}}}' \
http://localhost/vsaas/api/v2/my/subscriptions
```

Please note that the `mode` parameter can take one of the two values: `single` (one person) or `list` (list of persons); depending on that, `id` is the identifier of a person or a person list correspondingly.

In the `url` parameter, specify the address of the server to which the notifications should be sent. The `notification_type` supports only `http` at the moment.

6) Test how recognition works and make sure that the integration module is working correctly.

6.1.10 Auto-login

Flussonic Watcher allows its users to log in by a special URL (auto-login link), without entering a password. This is useful if you want to simplify access to Flussonic Watcher for your users or prevent the transfer of passwords to third parties.

The auto-login link is issued to an authorized client.

In order to generate a URL for auto-login, you will need to first request a token by using the link `/vsaas/api/v2/auth/generate-autologin-token`. After that, the user can be authorized by sending the token via the POST method to `/vsaas/autologin`.

Follow the steps:

1) First, you need to generate a token for the autologin of a particular user. To do this, make a POST request in the JSON format as follows:

```
curl --header "X-Vsaas-Api-Key: API_KEY" \
--header "Content-Type: application/json" \
--request POST --data '{"login": LOGIN, "valid_till": VALID_TILL, "lifetime": LIFETIME}' \
"http://watcher.com/vsaas/api/v2/auth/generate-autologin-token"
```

In the request, replace these placeholders with real values:

- `API_KEY` — the Watcher API key that can be found in the Watcher UI on the **Settings** page in **API key**. This key is sent in the HTTP header `X-Vsaas-Api-Key`.
- `LOGIN` — the login (the same as the email) of the user to whom you want to give access. Line. Required.
- `VALID_TILL` — the UTC time in seconds until this token is valid for autologin. Integer. Optional parameter.
- `LIFETIME` — the duration of the session that will be open at the user's automatic logging in, in seconds. Integer. Optional parameter.

Watcher sends a response in the JSON format:

```
{
    "autologin_token": "demo:1487258314:f8b1:b4bdaac58cbe94638e5b14a3728b8e6d633f3c6e",
    "success": true
}
```

The `autologin_token` field contains the token for the specified user.

2) The `autologin_token` received in the previous step can be used in POST requests to Flussonic Watcher. For example:

```
<form action="http://watcher.com/vsaas/autologin" method="POST">
  <input type="hidden" name="autologin_token" value="AUTOLOGIN_TOKEN" />
  <input type="submit" />
</form>
```

At the click on the submit button, the user will be logged in automatically into the Flussonic Watcher web interface.

6.1.11 Organization management API

You can use Organization management API requests to integrate Watcher with billing or to develop your own application/web interface for Watcher, etc. For example, if Provider creates Organizations, presets and users from the Watcher interface and assigns the subscriber as the owner of the Organization, then the billing will most likely need to request information about Organization settings that the subscriber has chosen. If billing is a control system, implement the requests for creating, changing and deleting Organizations, limiting DVR, the number of cameras and users in the Organization.

Please refer to Integration with existing billing system for details on cameras provisioning and billing integration.

This section provides examples of API requests for managing organizations. Click here to see the full list of response and request parameters.

You should use one of the keys for the API request authorization. The examples in this section are given with the x-vsaas-session key: this means that the user on whose behalf the requests are made must have the rights to perform the corresponding actions. Please refer to API request authorization ways for details on the key types.

Getting a list of Organizations

For example: if you created organizations from the Watcher interface and want to get a list of organizations and their parameters in billing, use the following query

GET YOUR_WATCHER_URL/vsaas/api/v2/organizations/

Example:

```
curl --request GET \
--url http://127.0.0.1/vsaas/api/v2/organizations/ \
--header 'x-vsaas-session: <vsaas_session>'
```

This response contains information on three Organizations: Default, HOME, Cafe

```
[
    {
        "mosaic_count": 4,
        "id": 6,
        "domain": {
            "id": 1,
            "title": "Flussonic Watcher"
        },
        "user_count": 2,
        "user_limit": 2000,
        "dvr_limit": 1000,
        "can_view_stats": true,
        "camera_count": 3,
        "can_edit_users": true,
        "camera_limit": 5000,
        "owner": {
            "id": 14,
            "login": "Subscriber 1"
        },
        "can_edit_cameras": true,
        "is_member": true,
        "is_default": false,
        "title": "Cafe"
    },
    {
        "mosaic_count": 0,
        "id": 1,
        "domain": {
            "id": 1,
            "title": "Flussonic Watcher"
        },
        "user_count": 2,
        "user_limit": 1000,
        "dvr_limit": 100,
        "can_view_stats": true,
        "camera_count": 1,
        "can_edit_users": true,
        "camera_limit": 1000,
        "owner": {
            "id": 1,
            "login": "admin"
        },
        "can_edit_cameras": true,
        "is_member": true,
        "is_default": true,
        "title": "Default"
    },
    {
        "mosaic_count": 0,
        "id": 3,
        "domain": {
            "id": 1,
            "title": "Flussonic Watcher"
        },
        "user_count": 3,
        "user_limit": 2000,
        "dvr_limit": 1000,
        "can_view_stats": false,
        "camera_count": 1,
        "can_edit_users": true,
        "camera_limit": 5000,
        "owner": {
            "id": 3,
            "login": "Subscriber 2"
        },
        "can_edit_cameras": true,
        "is_member": true,
        "is_default": false,
        "title": "HOME"
    }
]
```

Creating an Organization

Use the following request to create an Organization

POST YOUR_WATCHER_URL/vsaas/api/v2/organizations/

Request parameters:

- title (string) — required parameter, name of the created organization.
- camera_limit (integer) — limitation on the number of cameras. If the parameter is not specified or is equal to 0, then the default limit of 1000 is set.
- dvr_limit (integer) — limitation on the disk space. If the parameter is not specified, then the default limit of 100 GB will be set.
- owner_id (integer) — the identifier of the user who is the administrator (owner) of the Organization. If this parameter is not specified, then the organization will be created without an owner, and no one, even the Super Administrator, will have rights to edit the settings of cameras, users and mosaics in this organization. A user with Watcher Administrator or Super Administrator rights will be able to change the owner of the Organization. Please refer to Managing users and their permissions for details on the permissions mechanism.
- user_limit (integer) — limitation of the number of users. If the parameter is not specified, then the default value of 1000 will be set.

Example:

```
curl --request POST \
--url http://127.0.0.1/vsaas/api/v2/organizations \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"title": "Org2"
}'
```

```
{
    "id": 12,
    "domain": {
        "id": 1,
        "title": "Flussonic Watcher"
    },
    "user_limit": 1000,
    "dvr_limit": 100,
    "camera_limit": 1000,
    "owner": null,
    "title": "Org2"
}
```

Information about the Organization

Use the following request to get information about specific Organization

GET YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)

Example with ID 1

```
curl --request GET \
--url http://127.0.0.1/vsaas/api/v2/organizations/1 \
--header 'x-vsaas-session: <vsaas_session>'
```

```
{
    "mosaic_count": 0,
    "id": 1,
    "domain": {
        "id": 1,
        "title": "Flussonic Watcher"
    },
    "user_count": 2,
    "user_limit": 1000,
    "dvr_limit": 100,
    "can_view_stats": true,
    "camera_count": 1,
    "can_edit_users": true,
    "camera_limit": 1000,
    "owner": {
        "id": 1,
        "login": "admin"
    },
    "can_edit_cameras": true,
    "is_member": true,
    "is_default": true,
    "title": "Default"
}
```

Changing the Organization

Use the following request to change one or several parameters of the Organization:

PUT YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)

The request parameters are the same as in the request to create the Organization. You can include only the parameters you want to change in the request body.

Example: change the Organization with ID = 12

```
curl --request PUT \
--url http://127.0.0.1/vsaas/api/v2/organizations/12 \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"title": "test org 2"
}'
```

Watcher will return an updated list of Organization parameters, similar to the example above.

Deleting the Organization

Use the following request to delete the Organization

DELETE YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)

Please note that the 'force' header must be added to the request, otherwise the request will return an error. This is because any Organization has a default folder that cannot be deleted while the Organization cannot be deleted normally as long as it has dependent objects (i.e. folders).

Example:

```
curl --request DELETE \
--url http://127.0.0.1/vsaas/api/v2/organizations/1 \
--header 'x-vsaas-session: <vsaas_session>' \
--header 'force: 1'
```

In case of successful deletion, Watcher will respond with {"success":true}

6.1.12 Folder management API

You can use folder management API requests to integrate Watcher with billing or to develop your own application/web interface for Watcher, etc. Depending on the requirements for the integration/application, you may want to create, delete or change folders, change their hierarchy, find out which users have access to the folder and which cameras are added to it, which folders the user has access to. The Watcher API supports all of these requests.

See Adding Cameras to Folders or Favorites for details on folders. Please refer to Integration with existing billing system for details on cameras provisioning and billing integration.

This section provides examples of API requests for managing folders. Click here to see the full list of response and request parameters.

You should use one of the keys for the API request authorization. The examples in this section are given with the x-vsaas-session key: this means that the user on whose behalf the requests are made must have the rights to perform the corresponding actions. Please refer to API request authorization ways for details on the key types.

Getting a list of folders

To get a list of folders in a specific Organization, use the following query

GET YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/folders

If you include the current user's x-vsaas-session key in the header, the request will return a list of folders available to the current user. To get a list of all folders in an organization, use x-vsaas-api-key.

Any organization has a default root folder with the same name as Organization.

Example of requesting folders in an organization with ID 6

```
curl 'http://localhost/vsaas/api/v2/organizations/6/folders' \
-H 'x-vsaas-session: tPbYSDBiaX4C8_BMuySsr63r-qQ'
```

Two folders in the Organization: the default root folder "Cameras" and its subfolder "New folder" created by some user.

```
[
    {
        "map_topleft": null,
        "parent_id": null,
        "map_topright": null,
        "order_num": 1,
        "coordinates": null,
        "camera_count": 2,
        "map_bottomleft": null,
        "organization_id": 6,
        "id": 7,
        "title": "Cameras",
        "level": 0
    },
    {
        "map_topleft": null,
        "parent_id": 7,
        "map_topright": null,
        "order_num": 1,
        "coordinates": null,
        "camera_count": 1,
        "map_bottomleft": null,
        "organization_id": 6,
        "id": 18,
        "title": "New folder",
        "level": 1
    }
]
```

Creating a folder

Use the following request to create a folder

POST YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/folders

Required parameters:

- title (string) — folder name
- parent_id (integer) — parent folder ID.

Optional parameters:

- order_num (integer) — the sequential number of the folder in the list. If the parameter is not specified or is already in use, the next free sequence number is assigned.

If you want to create floor plans, use the following parameters:

- coordinates (string) — the image center coordinates, latitude and longitude are separated by a space, the decimal separator is a dot.
- map_bottomleft (string) — coordinates of the lower left corner of the image.
- map_topleft (string) — coordinates of the upper left corner of the image.
- map_topright (string) — coordinates of the upper right corner of the image.
- map_file (object) — floor plan image:
    - b64_content (string) — base64 encoded image
    - delete (boolean) — true to delete previously used image (if any), or false to keep it
    - mime_type (string) — MIME type of the image

Example

```
curl --request POST \
--url http://localhost/vsaas/api/v2/organizations/6/folders \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"title": "Folder name",
"parent_id": 7
}'
```

The request will return the parameters of the created folder. Note that the map_file object in the response contains only the url parameter.

```
{"map_topleft":null,
"parent_id":7,
"map_topright":null,
"map_file":{"url":null},
"order_num":4,
"coordinates":null,
"camera_count":0,
"map_bottomleft":null,
"organization_id":6,
"id":22,
"title":"Folder name",
"level":1}
```

Changing the folder

Use the following request to change one or several parameters of the folder

PUT YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/folders/(folder ID)

You can pass the same parameters as when creating a folder, as well as the following additional parameters that allow you to change the position of the folder in the tree:

move_after (integer) — the identifier of the folder to be followed by the current folder. move_before (integer) — the identifier of the folder before which the current folder is to be placed. move_inside (integer) — the identifier of the folder where the current folder should be placed.

You will find detailed examples of using these parameters below.

Example changing folder name

```
curl --request PUT \
--url http://localhost/vsaas/api/v2/organizations/1/folders/1/ \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"title": "toor"
}'
```

The request will return all the parameters of the changed object similar to the example above.

Changing the folder position in the tree:

The examples below are for the following folder tree in an Organization:

graph { node [shape=folder] "id=1" -- "id=2" "id=1" -- "id=3" "id=1" -- "id=4" }

MOVING A FOLDER

```
curl --request PUT \
--url http://localhost/vsaas/api/v2/organizations/1/folders/3/ \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"move_after": 4
}'
```

Result:

graph { node [shape=folder] "id=1" -- "id=2" "id=1" -- "id=4" "id=1" -- "id=3" }

CHANGING A FOLDER'S PARENT

```
curl --request PUT \
--url http://localhost/vsaas/api/v2/organizations/1/folders/3/ \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"move_inside": 4
}'
```

Result:

graph { node [shape=folder] "id=1" -- "id=2" "id=1" -- "id=4" "id=4" -- "id=3" }

Removing the folder

The request to remove the folder is:

DELETE YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/folders/(folder ID)

Example

```
curl --request DELETE \
--url 'http://localhost/vsaas/api/v2/organizations/1/folders/1' \
--header 'x-vsaas-session: <vsaas_session>'
```

In case of successful deletion, Watcher will respond with {"success":true}

Request for cameras in the folder

Use the following request to find out which cameras are added to the folder:

GET YOUR_WATCHER_URL/vsaas/api/v2/cameras?folder_id=(folder ID)

Example

```
curl --request GET \
--url 'http://localhost/vsaas/api/v2/cameras?folder_id=7' \
--header 'x-vsaas-session: <vsaas_session>'
```

The response will contain all parameters of all cameras added to the folder. Cameras from child folders are not included in the response.

In this case, the response contains two cameras:

```
[
    {
        "agent_key": null,
        "agent_id": null,
        "thumbnails_url": "",
        "motion_detector_enabled": false,
        "static": true,
        "onvif_profile": null,
        "groups": [],
        "precise_thumbnails": false,
        "enabled": true,
        "dvr_space": null,
        "folder_id": 7,
        "precise_thumbnails_days": 0,
        "dvr_lock_days": 1,
        "agent_model": null,
        "user_attributes": {},
        "playback_config": {
            "token": "2.-s9tHGUHAAEABcXhj_yu3vqQhCc5l9Zv1aS-onP8IcVoG4Pl"
        },
        "thumbnails": false,
        "folder_coordinates": null,
        "substream_url": "",
        "preset": {
            "vision_alg": null,
            "vision_params": {},
            "vision_areas": null,
            "is_adjustable": true,
            "vision_enabled": null,
            "dvr_depth": 2,
            "domain_id": 1,
            "dvr_space": null,
            "vision_gpu": null,
            "id": 8,
            "is_default": true,
            "dvr_lock_days": 1,
            "title": "2+1days_adjustable",
            "is_deleted": false
        },
        "comment": "",
        "vision_alg": null,
        "last_change": {
            "object_type": "Camera",
            "created_at": 1623922927,
            "object_id": "samplecamera2-1584a116d9",
            "user": "admin"
        },
        "streamer_id": 1,
        "onvif_ptz": false,
        "preset_id": 8,
        "dvr_path": "/dvr",
        "coordinates": "",
        "permissions": {
            "dvr": true,
            "ptz": true,
            "edit": true,
            "view": true,
            "actions": true
        },
        "postal_address": "",
        "agent_serial": null,
        "dvr_depth": 2,
        "organization_id": 6,
        "video_only": false,
        "external_id": null,
        "stream_url": "file://vod/sample-10s.mp4 ",
        "agent_status": null,
        "vision_areas": null,
        "onvif_url": null,
        "vision_enabled": null,
        "last_event_time": null,
        "stream_status": {
            "alive": false,
            "lifetime": 5948,
            "source_error": null,
            "http_port": 80,
            "https_port": null,
            "bitrate": 4294,
            "server": "localhost"
        },
        "name": "samplecamera2-1584a116d9",
        "has_actions": false,
        "vision_gpu": null,
        "title": "SampleCamera2"
    },
    {
        "agent_key": null,
        "agent_id": null,
        "thumbnails_url": "",
        "motion_detector_enabled": false,
        "static": true,
        "onvif_profile": null,
        "groups": [],
        "precise_thumbnails": false,
        "enabled": true,
        "dvr_space": null,
        "folder_id": 7,
        "precise_thumbnails_days": 0,
        "dvr_lock_days": 1,
        "agent_model": null,
        "user_attributes": {},
        "playback_config": {
            "token": "2.-s9tHGUHAAEABcXhj_yu3gCkwPvOTwRbY3O60MZ0Qb117UHv"
        },
        "thumbnails": false,
        "folder_coordinates": null,
        "substream_url": "",
        "preset": {
            "vision_alg": null,
            "vision_params": {},
            "vision_areas": null,
            "is_adjustable": false,
            "vision_enabled": null,
            "dvr_depth": 1,
            "domain_id": 1,
            "dvr_space": null,
            "vision_gpu": null,
            "id": 7,
            "is_default": true,
            "dvr_lock_days": 1,
            "title": "1+1days_non-adjustable",
            "is_deleted": false
        },
        "comment": "",
        "vision_alg": null,
        "last_change": {
            "object_type": "Camera",
            "created_at": 1623923311,
            "object_id": "samplecamera4-fd2f695a2e",
            "user": "admin"
        },
        "streamer_id": 1,
        "onvif_ptz": false,
        "preset_id": 7,
        "dvr_path": "/dvr",
        "coordinates": "",
        "permissions": {
            "dvr": true,
            "ptz": true,
            "edit": true,
            "view": true,
            "actions": true
        },
        "postal_address": "",
        "agent_serial": null,
        "dvr_depth": 1,
        "organization_id": 6,
        "video_only": false,
        "external_id": null,
        "stream_url": "file://vod/sample-5s.mp4 ",
        "agent_status": null,
        "vision_areas": null,
        "onvif_url": null,
        "vision_enabled": null,
        "last_event_time": null,
        "stream_status": {
            "alive": false,
            "lifetime": 6008,
            "source_error": null,
            "http_port": 80,
            "https_port": null,
            "bitrate": 4081,
            "server": "localhost"
        },
        "name": "samplecamera4-fd2f695a2e",
        "has_actions": false,
        "vision_gpu": null,
        "title": "SampleCamera4"
    }
]
```

Users in the folder

GETTING A LIST OF USERS IN THE FOLDER

Use the following query to get a list of users who have permission to access the folder

GET YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/folders/(folder ID)/users

```
curl --request GET \
--url http://localhost/vsaas/api/v2/organizations/1/folders/1/users \
--header 'x-vsaas-session: <vsaas_session>'
```

```
[
    {
        "can_use_ptz": true,
        "can_use_actions": true,
        "can_view_dvr": true,
        "user_id": 16,
        "dvr_depth_limit": null
    },
    {
        "can_use_ptz": false,
        "can_use_actions": false,
        "can_view_dvr": true,
        "user_id": 14,
        "dvr_depth_limit": 1
    }
]
```

VIEWING SPECIFIC USER'S PERMISSIONS IN ALL AVAILABLE FOLDERS

To get a list of user permissions in all folders available to them, use the query

GET YOUR_WATCHER_URL/vsaas/api/v2/users/(user_id)?fields=folders_permissions'

```
curl --request GET \
--url http://localhost/vsaas/api/v2/users/2?fields=folders_permissions' \
--header 'x-vsaas-session: <vsaas_session>'
```

```
{
    "folders_permissions": [
        {
            "folder_id": 1,
            "organization_id": 1,
            "can_view_dvr": true,
            "dvr_depth_limit": 365,
            "can_use_ptz": false,
            "can_use_actions": false
        },
        {
            "folder_id": 25,
            "organization_id": 12,
            "can_view_dvr": false,
            "dvr_depth_limit": null,
            "can_use_ptz": false,
            "can_use_actions": false
        }
    ]
}
```

VIEWING SPECIFIC USER'S PERMISSIONS IN THE FOLDERS OF SPECIFIC ORGANIZATION

To find out what permissions a user has to the folders of the Organization, use the following query:

GET YOUR_WATCHER_URL/vsaas/api/v2/users/(user_id)?fields=folders_permissions'

```
curl --request GET \
--url http://localhost/vsaas/api/v2/organizations/1/users/2?fields=folders_permissions' \
--header 'x-vsaas-session: <vsaas_session>'
```

```
{
    "folders_permissions": [
        {
            "folder_id": 1,
            "can_view_dvr": true,
            "dvr_depth_limit": 6,
            "can_use_ptz": false,
            "can_use_actions": false
        },
        {
            "folder_id": 13,
            "can_view_dvr": false,
            "dvr_depth_limit": null,
            "can_use_ptz": false,
            "can_use_actions": false
        }
    ]
}
```

ADDING THE USER TO THE FOLDER

Use the following request to give the user permissions to view live video or archive or control PTZ

POST YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/folders/(folder ID)/users

Request parameters:

- user_id (integer) — user's ID
- can_use_ptz (bool) — true to allow PTZ control, or false to forbid
- can_view_dvr (bool) — true to allow access to DVR, or false to forbid
- can_use_actions (bool) — true to allow actions management, or false to forbid
- dvr_depth_limit (integer) — the number of archive days available to the user; null stands for unlimited access

```
curl --request POST \
--url http://localhost/vsaas/api/v2/organizations/1/folders/1/users \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"can_view_dvr": 10,
"can_use_ptz": 1,
"user_id": 14
}'
```

```
{
    "can_use_actions": null,
    "dvr_depth_limit": null,
    "can_use_ptz": 1,
    "user_id": 14,
    "can_view_dvr": 10
}
```

CHANGING USER PERMISSIONS IN THE FOLDER

Use the following request to change the user's permissions to view live video and archive and control PTZ

POST YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/folders/(folder ID)/users

The request parameters are the same as when assigning permissions (see above).

```
curl --request PUT \
--url http://localhost/vsaas/api/v2/organizations/1/folders/1/users/1/ \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"can_use_ptz": 1,
"can_view_dvr": 5
}'
```

The request will return the user's permissions to the folder similar ro the example above.

DELETING THE USER FROM THE FOLDER

Use the following request to disable user access to the folder

DELETE https://YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization_id)/folders/(folder_id)/users/(user_id)

```
curl --request DELETE \
--url http://localhost/vsaas/api/v2/organizations/1/folders/1/users/1/ \
--header 'x-vsaas-session: <vsaas_session>' \
```

If the request is successful, Watcher will return {"success":true}.

6.1.13 User management API

You can use user management API requests to integrate Watcher with billing or to develop your own application/web interface for Watcher, etc. For example, if Provider creates Organizations, presets and users from the Watcher interface, then the billing will most likely need to request information about created users and which Organization(s) they are added to, user permissions assigned. If billing is a control system, implement the requests for creating, changing and deleting users as well as permissions assigning.

Please refer to Integration with existing billing system for details on cameras provisioning and billing integration. You can also find useful: Folder management API .

This section provides examples of API requests for managing users. Click here to see the full list of response and request parameters.

You should use one of the keys for the API request authorization. The examples in this section are given with the x-vsaas-session key: this means that the user on whose behalf the requests are made must have the rights to perform the corresponding actions. Please refer to API request authorization ways for details on the key types.

Creating a user in the Organization

If you have the list of users in csv, import all users at once as described in User import API.

If you are adding one user at a time, use this query

POST YOUR_WATCHER_URL/vsaas/api/v2/users?trace=sql

Required parameters:

- login (string) — the username to log into the Watcher app/web interface.
- password (string) — user password

Optional parameters for settings on the **Parameters** tab in the interface (see Managing users and their permissions for details):

- organization_id (integer) — ID of the Organization to add the user to. If not specified, the user will be added to the Default Organization. You can add the user into one Organization at creation. If you want to add the user to several Organizations, use the adding user request shown below after creation.
- can_edit_organizations (boolean) — true if the user can edit Organizations, default is false.
- can_view_organizations (boolean) — true if the user can view Organizations, default is false.
- enabled (boolean) — whether the user is active or not, default is true
- is_admin (boolean) — whether the user is Watcher Administrator, default is false.
- readonly (boolean) — enables read-only mode for the user, default is false.
- max_sessions (integer) — maximum number of sessions.
- can_edit_settings (boolean) — whether the user can manage Watcher settings (on the **Settings** page), default is false. There is no corresponding setting in the interface.

Other optional parameters:

- apikey (string) — user's API key which is one of the ways to authorize an API request. See API request authorization ways for details.
- locale (string) — interface language
- name (string) — user name
- note (string) — note
- notification_email (string (email)) — user's e-mail address
- phone (string) — user's phone number

Example

```
curl --request POST \
--url 'http://localhost/vsaas/api/v2/users?trace=sql' \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"login": "test",
"password": "test",
"organization_id": 1
}'
```

The response will contain all the parameters of the created user.

Please note that the Organizations array is returned in the response.

```
{
    "apikey": null,
    "id": 17,
    "can_view_organizations": false,
    "max_sessions": null,
    "name": null,
    "locale": "",
    "login": "test",
    "external_id": null,
    "organizations": [
        1
    ],
    "readonly": false,
    "can_edit_organizations": false,
    "note": null,
    "is_admin": false,
    "authorized_ip": null,
    "phone": null,
    "notification_email": null,
    "monitoring": false,
    "enabled": true,
    "can_edit_settings": false,
    "domain_id": 1
}
```

Getting a list of users in the Organization

Use the following request to get the list of users in the Organization

GET YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/users

Example

```
curl --request GET \
--url http://localhost/vsaas/api/v2/organizations/1/users \
--header 'x-vsaas-session: <vsaas_session>'
```

The response will contain some general parameters of users and their permissions within the Organization.

```
[
    {
        "notification_email": null,
        "can_edit_organization_users": true,
        "login": "Subscriber 1",
        "can_edit_organization_plists": true,
        "can_view_organization_stats": true,
        "can_edit_organization_cameras": true,
        "can_view_organization_plists": true,
        "user_id": 14,
        "folders_permissions": [
            {
                "folder_id": 1,
                "can_view_dvr": true,
                "dvr_depth_limit": 365,
                "can_use_ptz": false,
                "can_use_actions": false
            },
            {
                "folder_id": 25,
                "can_view_dvr": false,
                "dvr_depth_limit": null,
                "can_use_ptz": false,
                "can_use_actions": false
            }
        ]
    },
    {
        "notification_email": null,
        "can_edit_organization_users": false,
        "login": "Subscriber 2",
        "can_edit_organization_plists": false,
        "can_view_organization_stats": false,
        "can_edit_organization_cameras": false,
        "can_view_organization_plists": false,
        "user_id": 24,
        "folders_permissions": []
    }
]
```

Adding the user to the Organization

Use the following request to add an existing user to the Organization

POST YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/users

Request parameters:

- user_id (integer) — required parameter, identifier of the added user
- login (string) — username
- notification_email (string) — user's e-mail address
- can_view_organization_plists (boolean) — permissions to view the person lists within Organization
- can_edit_organization_plists (boolean) — permissions to edit the person lists within Organization
- can_edit_organization_users (boolean) — permissions to edit users
- can_edit_organization_cameras (boolean) — permissions to edit cameras within the Organization
- can_view_organization_stats (boolean) — permissions to view statistics
- folders_permissions (array) — a list of the Organization's folders to which the user has access

Example

```
curl --request POST \
--url http://localhost/vsaas/api/v2/organizations/1/users \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data ' {
"user_id": 2,
"can_edit_organization_cameras": true
}'
```

The response will contain some general parameters of the user and their permissions within the Organization.

```
{
    "can_edit_organization_plists": false,
    "can_edit_organization_users": false,
    "user_id": 2,
    "notification_email": null,
    "can_edit_organization_cameras": true,
    "can_view_organization_stats": false,
    "folders_permissions": [],
    "can_view_organization_plists": false,
    "login": "test"
}
```

Getting permissions of one user

Use the following request to get the user's permissions

GET YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/users/(user ID)

Example

```
curl --request GET \
--url http://localhost/vsaas/api/v2/organizations/1/users/2/ \
--header 'x-vsaas-session: <vsaas_session>'
```

The response will contain some general parameters of the user and their permissions within the Organization similar to the example above.

Changing the user's permissions within the Organization

Use the following request to change the user's permissions within the Organization

PUT YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/users/(user ID)

The request parameters are the same as when adding an existing user to the Organization. You can include only those parameters that need to be changed to the request body.

Example

```
curl --request PUT \
--url http://localhost/vsaas/api/v2/organizations/1/users/2/ \
--header 'content-type: application/json' \
--header 'x-vsaas-session: <vsaas_session>' \
--data '{
"can_view_organization_stats": true,
"can_edit_organization_users": true,
"can_edit_organization_cameras": true
}'
```

The response will contain some general parameters of the user and their permissions within the Organization similar to the example above.

Deleting the user from the Organization

Use the following request to delete the user from the Organization

DELETE YOUR_WATCHER_URL/vsaas/api/v2/organizations/(organization ID)/users/(user ID)

```
curl --request DELETE \
--url http://localhost/vsaas/api/v2/organizations/1/users/2/ \
--header 'x-vsaas-session: <vsaas_session>'
```

If the deletion is successful, Watcher responds with {"success":true}