AltumView Developer FAQ

Last updated: May 6, 2025

Table of Contents

Introduction

The AltumView API uses REST (REpresentational State Transfer) with the JSON format. REST is a web-based architecture and uses HTTP protocols. It revolves around the concept of resource, where every component is a resource and a resource is accessed by common interface using HTTP standard methods: GET, POST, DELETE, PATCH.

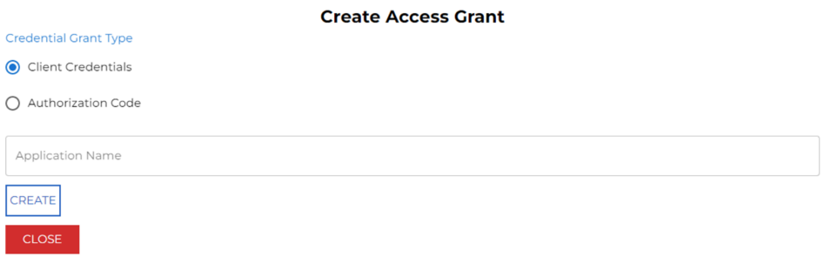
Postman

To get started quickly, download the Postman collection for the server you're using, and import it to your Postman application. Read the Overview page within the Postman collection carefully, as it provides important instructions, such as how to retrieve your own Client ID and Client Secret to fetch an access token (also discussed in this FAQ).

- **USA**: Altumview API - USA - Client Credentials.postman_collection.json
- **Canada**: Altumview API - Canada - Client Credentials.postman_collection.json
- **China**: Altumview API - China - Client Credentials.postman_collection.json
- **Europe**: Altumview API - Europe - Client Credentials.postman_collection.json

1) How do I get started?

1. Request API access from us and provide the email of your Sentinare app account. By default, API access is not enabled for new accounts.
2. Go to one of the following URLs, depending on which regional server you are using.
   - **USA**: https://accounts.altumview.com
   - **Canada**: https://accounts.altumview.ca
   - **China**: https://accounts.altumview.com.cn
   - **Europe**: https://accounts.altumview.co
3. Log in with your Sentinare app account.
4. Using the menu on the left corner of the screen, navigate to the "OAuth 2.0 Credentials" page.
   - Note: if you don't see the menu on the top left and we've already enabled API access for you, log out then log back in.
5. Click the CREATE GRANT button near the upper right corner. We provide two types of access grant flows, client_credentials flow and authorization_code flow. Please refer to Section 3 below for details.



Create Access Grant

Credential Grant Type

● Client Credentials

○ Authorization Code

Application Name

2) What are the API document links and the API URLs?

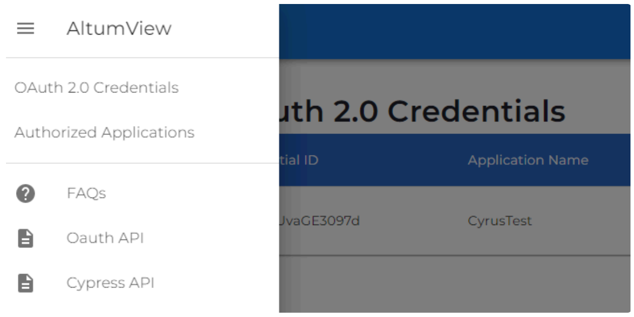
The AltumView API document can be found at

https://docs.altumview.com/cypress_api/

The AltumView OAuth API document can be found at

https://docs.altumview.com/oauth_api/

These API documents and this FAQ document can also be accessed by clicking the menu icon in the upper left corner of your account page in Section 1 above.



Important Note:

<u>The examples in the API documents above always use the USA server's URLs.</u> If you are using other servers, please change the USA URLs to your server's URLs using the following tables, otherwise your access will be denied.

USA (Default)	
OAuth API URL	https://oauth.altumview.com/v1.0
AltumView API URL	https://api.altumview.com/v1.0
MQTT Host URL	prod.altumview.com

Canada	
OAuth API URL	https://oauth.altumview.ca/v1.0
AltumView API URL	https://api.altumview.ca/v1.0
MQTT Host URL	prodca.altumview.ca

China	
OAuth API URL	https://oauth.ailecare.cn/v1.0
AltumView API URL	https://api.ailecare.cn/v1.0
MQTT Host URL	beijing.altumview.com.cn

Europe	
OAuth API URL	https://oauth.altumview.co/v1.0
AltumView API URL	https://api.altumview.co/v1.0
MQTT Host URL	ireland.altumview.co

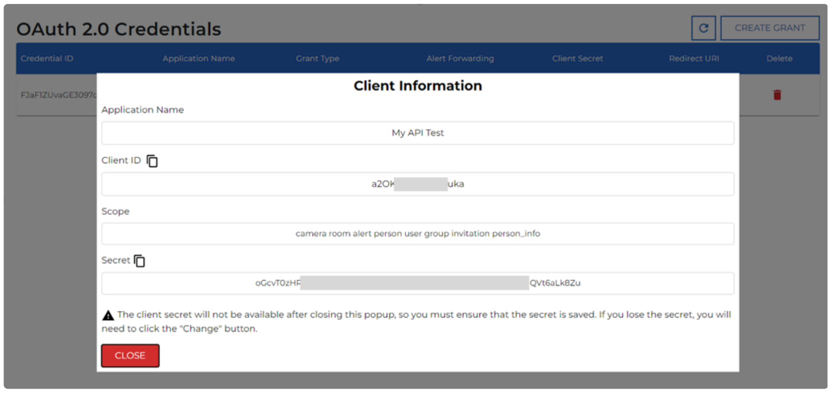
---

3) How do I get an access token?

You can get the access token from our OAuth server. We provide two types of grant flows, client_credentials flow and authorization_code flow. Please choose the suitable approach based on your application.

Method A) client_credentials:

This method is simpler and can be used if you just want to have one account where your system manages all the sensors and people (for yourself and for your users) through one access token. If you choose this method from your Account page in Sec. 1 above, we will generate a Client ID (`client_id`) and a Secret (`client_secret`) for you, as shown in the following figure. Please keep these values securely. If they have been compromised, you can change them by clicking the Change button under Client Secret in your account, as shown below.



Credential ID	Application Name	Grant Type	Alert Forwarding	Client Secret	Redirect URI	Delete
FJaF1ZUvaGE3097c						

Client Information

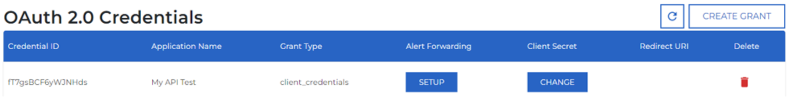
Application Name: My API Test

Client ID: a2Ok[redacted]uka

Scope: camera room alert person user group invitation person_info

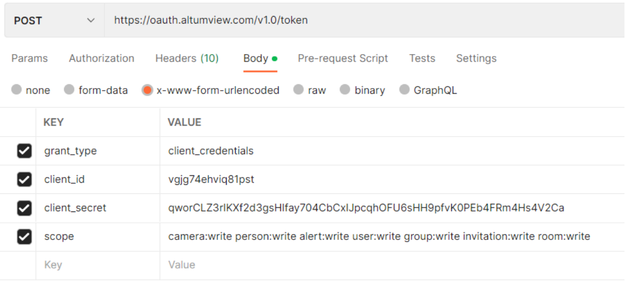
Secret: oGcvT0zHF[redacted]QVt6aLk8Zu

⚠ The client secret will not be available after closing this popup, so you must ensure that the secret is saved. If you lose the secret, you will need to click the "Change" button.



Credential ID	Application Name	Grant Type	Alert Forwarding	Client Secret	Redirect URI	Delete
fT7gsBCF6yWJNHds	My API Test	client_credentials	SETUP	CHANGE		

To retrieve an acces token, use the Oauth **[POST /token]** endpoint with **x-www-form-urlencoded** body:


POST https://oauth.altumview.com/v1.0/token

○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	KEY	VALUE
☑	grant_type	client_credentials
☑	client_id	vgjg74ehviq81pst
☑	client_secret	qworCLZ3rlKXf2d3gsHlfay704CbCxlJpcqhOFU6sHH9pfvK0PEb4FRm4Hs4V2Ca
☑	scope	camera:write person:write alert:write user:write group:write invitation:write room:write
	Key	Value

The **scope** parameter defines what resources your access_token can read or write. Check the API documentation to see which ones you will need first. You can refer to the scope option from permission field of the AltumView API documentation.

The main/admin account, which is created through the Sentinare app or on Sentinare Web, has the following scopes:

```
camera:write person:write alert:write user:write group:write
invitation:write room:write arn:write skeleton:read
```

`alexa:write person_info:write`

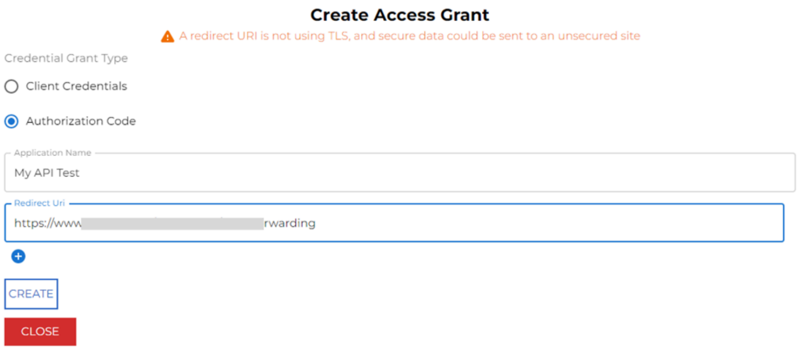
A secondary user, which is a type of user invited by the admin account, has the following scopes:

`camera:read person:read alert:write user:write group:read invitation:read room:read arn:write skeleton:read alexa:write person_info:write`

Method B) authorization_code:

This method is more advanced, and should be used if you plan to have your users create their own accounts (where they add Sentinare sensors and people within their own account). In this case, they will be registering Sentinare accounts and authenticating through our OAuth Webpage. Each user will request their own access token for accessing their own resources through our API.

If you choose this method, you need to provide a redirect URL, as shown below.



Create Access Grant

⚠ A redirect URI is not using TLS, and secure data could be sent to an unsecured site

Credential Grant Type

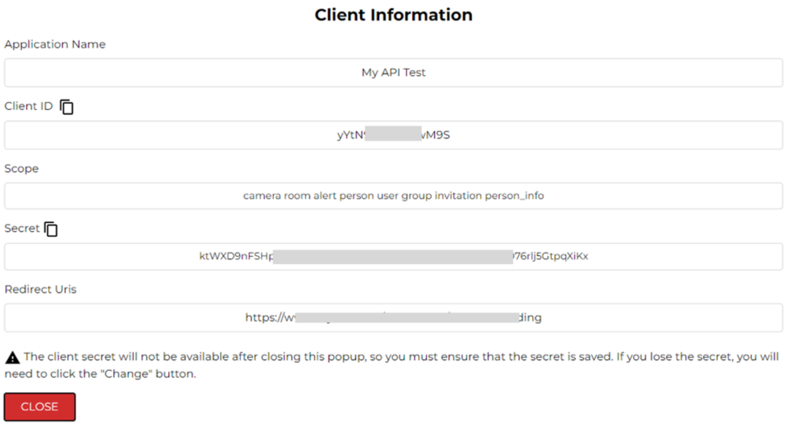
○ Client Credentials

● Authorization Code

Application Name: My API Test

Redirect Uri: https://www[redacted]rwarding

After clicking Create, we will generate a Client ID (`client_id`) and a Secret (`client_secret`) for you, as shown in the following screenshot.



Client Information

Application Name

My API Test

Client ID

yYtN[redacted]vM9S

Scope

camera room alert person user group invitation person_info

Secret

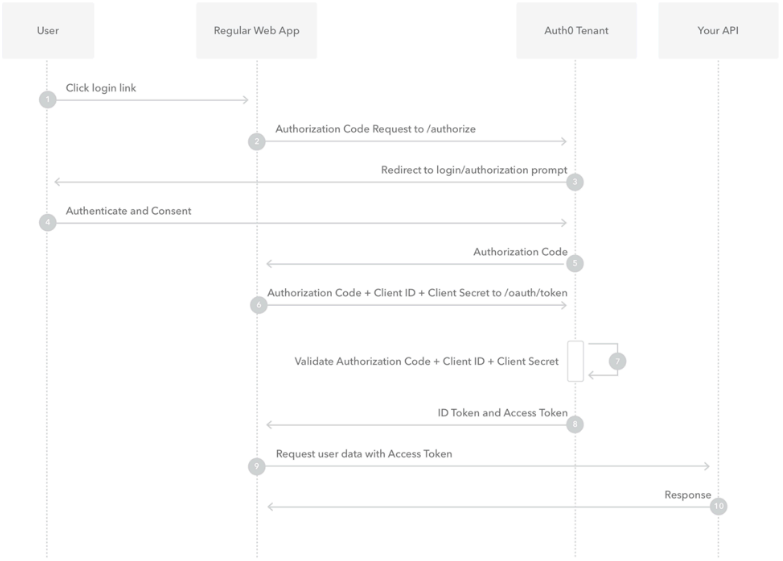
ktWXD9nFSHp[redacted]76rlj5GtpqXiKx

Redirect Uris

https://w[redacted]ding

⚠ The client secret will not be available after closing this popup, so you must ensure that the secret is saved. If you lose the secret, you will need to click the "Change" button.

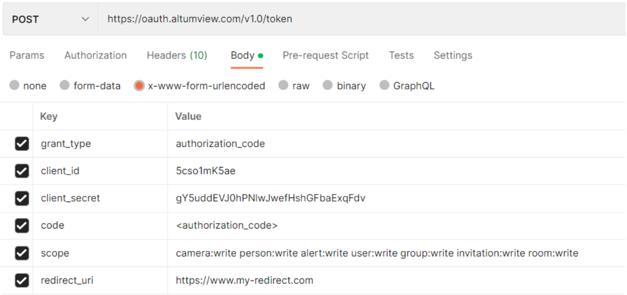
The authorization code grant type is a method defined in the OAuth standard. The following screenshot shows how it works.


User
Regular Web App
Auth0 Tenant
Your API
1 Click login link
2 Authorization Code Request to /authorize
3 Redirect to login/authorization prompt
4 Authenticate and Consent
5 Authorization Code
6 Authorization Code + Client ID + Client Secret to /oauth/token
7 Validate Authorization Code + Client ID + Client Secret
8 ID Token and Access Token
9 Request user data with Access Token
10 Response

In general, you'll need to:

- Create a button that calls the OAuth **GET /authorize** endpoint
  - The `redirect_uri` parameter should **not** be url-encoded
  - This endpoint will return an AltumView Login URL
  - PKCE is optional, but highly recommended. If you're unfamiliar with it, you can learn more about it and its purpose here: What the heck is PKCE?
    - We only accept the S256 challenge code method

- Once you receive the AltumView Login URL, open it in a browser. In other words, when your user clicks on the button, they will be taken to authenticate in a browser.
- After the user successfully authenticates, they will be redirected to the `redirect_uri` you've defined in your Accounts page, with an `authorization_code` parameter appended to this URL
- Have your system parse the URL for the `authorization_code`
- Complete the retrieval of the access token by calling the OAuth **[POST /token]** endpoint with **x-www-form-urlencoded** body, as shown in the screenshot below
- For example code of a web page using authorization code flow, please see our **[Authorization Code Demo.]**



POST https://oauth.altumview.com/v1.0/token

● x-www-form-urlencoded

	Key	Value
☑	grant_type	authorization_code
☑	client_id	5cso1mK5ae
☑	client_secret	gY5uddEVJ0hPNlwJwefHshGFbaExqFdv
☑	code	<authorization_code>
☑	scope	camera:write person:write alert:write user:write group:write invitation:write room:write
☑	redirect_uri	https://www.my-redirect.com

- **Important security notes:**
    - When calling the **[POST /token]** endpoint, `client_secret` is optional. It should only be included in confidential applications, but not in public applications. A confidential application is one that runs server-side and have its code hidden. On the other hand, a public application runs on the client's device, such as a mobile application or a web browser application, where the code and the client secret can be retrieved
    - The `client_secret` must be provided in order to retrieve a `refresh_token`
    - Therefore, you should consider hosting a proxy server, also referred to as a Backend for Frontend (BFF) service, that stores the `client_secret` and handles the token exchange. This helps prevent malicious users from easily stealing your client secret, and creating their own application that pretends to be yours

---

4) How do I use the access token?

As in standard OAuth 2.0 protocol, the client must send the access token in the `Authorization` header when making requests to protected resources. Do not forget the string “Bearer “ with the space and capital B, as these are common mistakes.

```
Authorization: Bearer <access_token>
```

5) How long does the access token last?

Use of refresh tokens are not needed for the client_credentials method, as you just need to make the client credentials API call (FAQ Section 3A) again if the token expires.

The access tokens generated using the authorization_code method are short-lived, so use the `expires_in` field to handle access token expiration, and make use of the `refresh_token` by calling the Oauth **POST /token** endpoint with grant_type: refresh_token to generate a new access token. Refresh tokens last until the grant is revoked in your Accounts page.

6) How do I get person activities data?

The activities data is inside GetPeopleById endpoint, under activities attribute. You will need to provide a **startDate** and **endDate** in epoch (seconds).

The activities is encoded in run length encoding, please deserialize the encoded string based on this method. Each number preceding a letter indicates the number of **minutes** spent on an activity, and each letter is denoted as such:

A: `idleOrUndetected`

B: `bending`

C: `lying`

D: `sittingChair`

E: `sittingFloor`

F: `squatting`

G: `standing`

H: `struggling`

7) Can I export the person activities data as csv or txt format?

Base on the security consent, you will need to collect the data with access token from our API. You can call our GetPeopleById endpoint with the `startDate` and `endDate` parameters of the desired time period

---

8) How do I get multiple people's activities at once?

We currently do not have the API for this feature. Please use the single GetPeopleById endpoint to retrieve the data iteratively.

---

9) How do I get real-time streaming skeleton data?

The steps to stream skeleton from an AltumView sensor are as follows, involving both HTTP requests and MQTT requests.

HTTP requests, as documented here: https://docs.altumview.com/cypress_api/#api-Utils-GetMqtt

1. Call the {GET} "cameras/:id/background" endpoint endpoint to retrieve the background image
2. Call the {GET} "/mqttAccount" endpoint to retrieve your username, password, and WSS URL. **Note:** You need to do this periodically as each MQTT session only lasts a short period of time, as indicated by the `expires_at` field. **Furthermore, each user account may only request up to 20 sessions at a time. See Section 12 for how to avoid running out of the sessions.**
3. Call the {GET} "/streamtoken" endpoint to retrieve your sensor's stream token. The streamToken changes every 24 hours
4. Call the {GET} "/info" endpoint to get your Group ID once. This ID never changes, so cache it for future use

MQTT requests:

1. Log into WebSocket over MQTT using the username, password, and WSS URL
2. Every 45 seconds, publish to this MQTT topic with streamToken as the message payload: **mobile/${groupId}/camera/${serialNumber}/token/mobileStreamToken**

   - **IMPORTANT**: even though you publish this every 45 seconds, do not fetch a new streamToken every time to strain bandwidth. The streamToken only changes every 24 hours

3. Subscribe to this MQTT topic to get the skeleton data **mobileClient/${groupId}/camera/${serialNumber}/skeleton/${streamToken}**

The attached file, ***StreamDemo.html***, is a Javascript demo of the MQTT connection, binary data parsing and rendering.

https://docs.altumview.com/resources/StreamDemo.html

The link does not work because you must configure the parameters based on your own information:

- client credentials (refer to FAQ section 1)
- server URLs (refer to FAQ section 2)
- sensor serial number (retrieve from the app)
- camera ID (call the {GET} **/cameras** endpoint)
- sensor stream token (call the {GET} **/cameras/:id/streamToken** endpoint)
- account group ID (call the {GET} **/info** endpoint)

Make the changes in the code, then open the file in a browser to run it, and check the process in the browser's developer console.

The attached PDF, ***SH-RealTimeSkeletonBinaryFormat***, outlines the format of the binary. The MQTT message payload is a bytes array, and you will need to process them accordingly. The attached *RealTimeSkeletonProcessor.cs* file is a C# example of processing the data into one frame.

https://docs.altumview.com/resources/SH-RealTimeSkeletonBinaryFormat.pdf

https://docs.altumview.com/resources/RealTimeSkeletonProcessor.cs

Important Notes:

- The coordinates are of type FLOAT and are normalized between 0 and 1. So if you are able to process the data and receive all 18 X-Y coordinates between 0 and 1, great!

- The data does not contain width, height, or aspect ratio, you must **assume the aspect ratio is 16:9**
- When all skeletons have left the scene, the camera does not send an empty frame to indicate it's time to clear the last skeleton off the screen. You have to implement your own logic to clear the screen if you do not receive a new frame after some time (e.g. 2 seconds).

---

10) Why is the image flipped horizontally?

Yes, the output of the sensor is flipped relative to our perspective, so you will need to manually flip it horizontally for both the coordinates and background image.

---

11) I was getting streaming skeleton data, but then it stopped.

There are two common reasons why streaming could stop:

1. As mentioned, you will need to publish the stream token every 45 seconds to keep the streaming going. However, you do not need to retrieve a new streamToken every time, as that only expires every 24 hours.
2. The MQTT session expires periodically, based on the `expires_at` field, so you will need retrieve a new MQTT account before it expires.

---

12) I'm getting an error when requesting a new MQTT account for streaming.

To reduce the workload of the server, each account may only call the GET /mqttAccount endpoint 20 times to create **up to 20 active MQTT sessions at a time**. After that, you may only create another MQTT session when one or more of the previous sessions have expired. The following pointers may be helpful:

1. **One MQTT session can subscribe to multiple sensor stream topics**. For example, to stream from 6 sensors, you should subscribe to the stream topics of the 6 sensors. You do not need 6 MQTT sessions. This can reduce the number of MQTT sessions you use and avoid hitting the 20-session limit. Note that the streaming may be suspended if too much data is streamed. **To reduce data usage, always remember to unsubscribe from any unused topic.**
2. **Only one application client may connect to a MQTT session**. This means that one MQTT session can support only one instance of an application, no matter whether it's a mobile

app or a web app, and our API only allows up to 20 instances (for example, 12 mobile app users + 8 web app users) to connect to MQTT.

3. **Each app should call {GET} /mqttAccount only once, caching the session credential and reusing it before it expires.** Do not place the calling of {GET} /mqttAccount in a retry logic. For example, if the app failed to stream from a sensor that is offline, do not create a new MQTT session. Doing so will not help.

After a session has expired, you may request a new one again as long as you have not exceeded the 20. However, note that you cannot share the same session credential among multiple clients, as doing so will kick the previous client off MQTT.

If you have a scenario in which you need more, please contact us with your use-case and account's email to discuss raising this limit. Additional fee may be charged.

---

13) How do I get the skeleton recordings?

The skeleton recordings data can be retrieved via the GetRecordings endpoint and the GetRecordingById endpoint.

Once you retrieve the data, refer to the following document to understand the binary format. Note that the format similar to that of the skeleton stream data and the alert playback in some ways, but it is still different. We have implemented more space-saving techniques to minimize the amount of space that recordings take up.

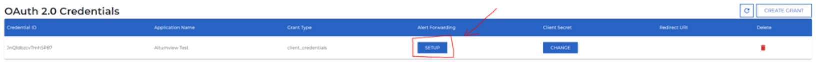
https://docs.altumview.com/resources/SH-SkeletonRecordingsBinaryFormat.pdf

---

14) Can I get the alert data forwarded to my server?

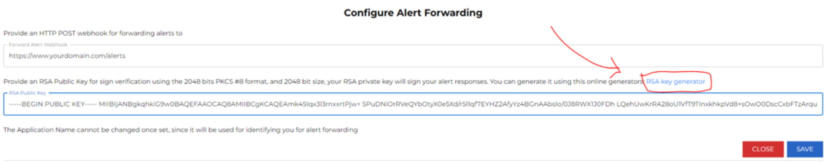
Yes. In your Accounts page (FAQ Section 1), you can setup Alert Forwarding after you have created a grant, where there is also an Alert Forwarding Demo (zip) for your reference.

After created the grant, click on "Setup" on Alert Forwarding. And, set up your POST endpoint URL and the public key.



OAuth 2.0 Credentials

Credential ID	Application Name	Grant Type	Alert Forwarding	Client Secret	Redirect URI	Delete
JnQ1dbzcv7mhSP87	Altumview Test	client_credentials	SETUP	CHANGE		🗑



Configure Alert Forwarding

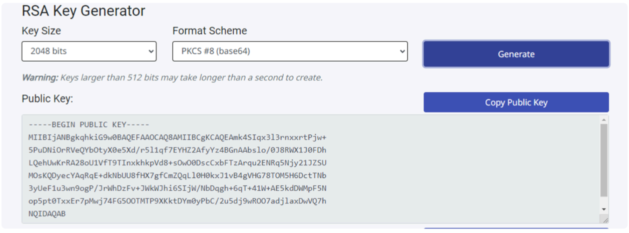
Provide an HTTP POST webhook for forwarding alerts to

Forward Alert Webhook: https://www.yourdomain.com/alerts

Provide an RSA Public Key for sign verification using the 2048 bits PKCS #8 format, and 2048 bit size, your RSA private key will sign your alert responses. You can generate it using this online generators RSA key generator

RSA Public Key: -----BEGIN PUBLIC KEY----- MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAmk4SIqx3l3rnxxrtPjw+ 5PuDNiOrRVeQYbOtyX0e5Xd/r5l1qf7EYHZ2AfyYz4BGnAAbslo/0J8RWX1J0FDh LQehUwKrRA28oU1VfT9TInxkhkpVd8+sOwO0DscCxbFTzArqu

The Application Name cannot be changed once set, since it will be used for identifying you for alert forwarding



RSA Key Generator

Key Size: 2048 bits

Format Scheme: PKCS #8 (base64)

Warning: Keys larger than 512 bits may take longer than a second to create.

Public Key:

```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAmk4SIqx3l3rnxxrtPjw+
5PuDNiOrRVeQYbOtyX0e5Xd/r5l1qf7EYHZ2AfyYz4BGnAAbslo/0J8RWX1J0FDh
LQehUwKrRA28oU1VfT9TInxkhkpVd8+sOwO0DscCxbFTzArqu2ENRq5Njy21JZSU
MOsKQDyecYAqRqE+dkNbUU8fHX7gfCmZQqLl0H0kxJ1vB4gVHG78TOM5H6DctTNb
3yUeF1u3wn9ogP/JrWhDzFv+JWkWJhi6SIjW/NbDqgh+6qT+41W+AE5kdDWMpF5N
op5pt0TxxEr7pMwj74FG5OOTMTP9XKktDYm0yPbC/2u5dj9wRO07adjlaxDwVQ7h
NQIDAQAB
```

Before continue on next part, we suggests you to read and understand about AES encryption:

An Introduction to the Advanced Encryption Standard (AES)

了解 AES 加密算法

The attached file, ***AlertForwardingSetup.pdf***, is a detailed instruction to help you setup alert forwarding.

https://docs.altumview.com/resources/AlertForwardingSetup.pdf

After successful setup, the data will forwarded to your webhook in real-time.

15) How do I get the skeleton data from alerts for playback?

If we are forwarding the alert to your server, then you will take the provided Alert ID, and use the GetAlertById endpoint, and provide the ID of the forwarded alert.

If you are **not** using the forwarded alert approach, then you can first get a list of all alerts you received in the past, using the GetAlerts endpoint. This will give you a list of the Alert IDs, so you can then call the GET alert by ID endpoint.

16) How do I render the skeleton_file received from the GetAlertById endpoint?

Once you've called the GET alert by ID endpoint, take the skeleton_file field, which is encoded in Base64 format, and convert it to a bytes array. Check the binary alert format document for usage.

https://docs.altumview.com/resources/SH-alert_binary_format_v3.pdf

Please note that the format is completely different from the real-time stream data. For example, the coordinates are in fixed point fractional representations. You will need to check the documentation carefully. Also, the frames are in reversed order.

The attached file, ***AlertDemo.html***, is a Javascript demo of the HTTP request, base64 conversion to byte array, data parsing and rendering.

https://docs.altumview.com/resources/AlertDemo.html

The link does not work because you must configure the parameters based on your own information:

- client ID and secret (refer to FAQ Section 1)
- server URLs (refer to FAQ Section 2)
- alert ID (call the **GET /alerts** endpoint, and choose one alert)

Make the changes in the code, then open the file in a browser to run it, and check the process in the browser's developer console.

---

17) How do I get person gait analysis?

You can get the gait analysis data from the GetPeople endpoint and the GetPeopleById endpoint.

---

18) What is the code 28 error? How do I handle it?

If you get an error with code 28, it means that the access token used for the API call is no longer valid, for likely 2 reasons:

1. The access token has expired and was not refreshed. In this case, refer to question 5 of this FAQ to learn how to refresh the access token.
2. The account password was recently changed, thus all current sessions are revoked for security reasons.

For these two reasons, you must handle the automatic renewal of access tokens via the refresh token.

If the access token has not expired and the user receives a code 28 error, you must force them to logout and re-login to get a new access token and refresh token pair. You may explain that their session has been revoked, possibly due to a recent password change.

---

19) What are the limits of OAuth and Cypress API endpoints?

To maintain the reliability and security of our services, we limit the number of API requests within a given time period based on caller's IP address. The current limit for the OAuth API is 100 requests within 5 minutes, and the limit for the Cypress API is 1000 requests within 5 minutes. **If you exceed the limit, you will receive the Code 403 "Forbidden" error message, and your IP will be blocked! In this case, you need to wait for 5 minutes before calling the API again, otherwise your IP could continue to be blocked.** To avoid the code 403 "Forbidden" error, you need to monitor your number of calls to the APIs and make sure it does not exceed the limit. This also increases the reliability and security of your system if your system is attacked. In particular, do not call the APIs repeatedly without any limit. For example, if the app failed to stream from a sensor that is offline, do not call {GET} /mqttAccount in an infinite retry logic. You should stop after a few retries.

---

20) How do I get or submit a floor mask?

To get the current floor mask used by the camera, or to submit a custom one, refer to the document:

https://docs.altumview.com/resources/APIs-FloorMask.pdf

---

21) How do I get or submit the region of interest mask?

To get the current region of interest mask used by the camera, or to submit a custom one, refer to the document:

https://docs.altumview.com/resources/APIs-RegionOfInterestMask.pdf

---

22) How do I setup 2-way audio calls with a sensor?

To establish a 2-way audio call with a Sentinare 2 (or newer) sensor, refer to the document:

https://docs.altumview.com/resources/APIs-SIP.pdf

---

23) How do I setup a sensor via Bluetooth API, without the Sentinare app?

To setup a sensor using our Bluetooth API, without the Sentinare mobile app, refer to the document:

https://docs.altumview.com/resources/APIs-Bluetooth Communication.pdf