

SENQUIP API DATA ACCESS WITH PYTHON

1. Introduction

The Senquip Cloud API allows you to seamlessly integrate with the Senquip platform, providing real-time access to asset data and to streamline the management of multiple devices from a central platform. Whether you need to track performance, manage diagnostics, or leverage advanced analytics, this RESTful API offers a straightforward way to interact with our cloud services.

Key features include:

- **Real-Time Data:** Get up-to-date information on asset status and location.
- **Device Management:** Modify device settings, network configurations, and other parameters directly via the API.
- **Provisioning:** Add new devices to your account or perform bulk updates.

This application note provides simple Python examples for retrieving device data from the [Senquip Cloud API V2](#). Two approaches are included:

- **JSON output** for developers who want to integrate Senquip data directly into their applications, dashboards, or analytics pipelines.
- **CSV output** for engineers and analysts who prefer working with data in spreadsheets or other tools.

Both examples demonstrate how to authenticate, query a device, and return measurement data. They are intended as quick start guides that can be adapted to your own use case, whether for lightweight data collection, integration into larger systems, or offline analysis.

To access the Senquip API, you must have an account on the Senquip Portal. If you do not have one, create one for free, at portal.senquip.com.

To interact with devices via the API, the device must either have:

- **Trial Access:** Available for 90 days from first activation.
- **Hosted Plan:** Please find plan information on the Senquip Portal.

Document Number APN0045	Revision 1.0	Prepared By NGB	Approved By NB
Title Senquip API Data Access with Python			Page 2 of 8

2. Senquip API V2 Basics

For more information on working with the API, see the Senquip API V2 [documentation](#).

2.1. Authorization

All API requests must contain a valid API Key. To generate API credentials, under account settings, navigate to the *API Access* tab. Credentials can be generated using the *Show API Credentials* button.

Copy the API Key value. This is your access token, keep it safe!

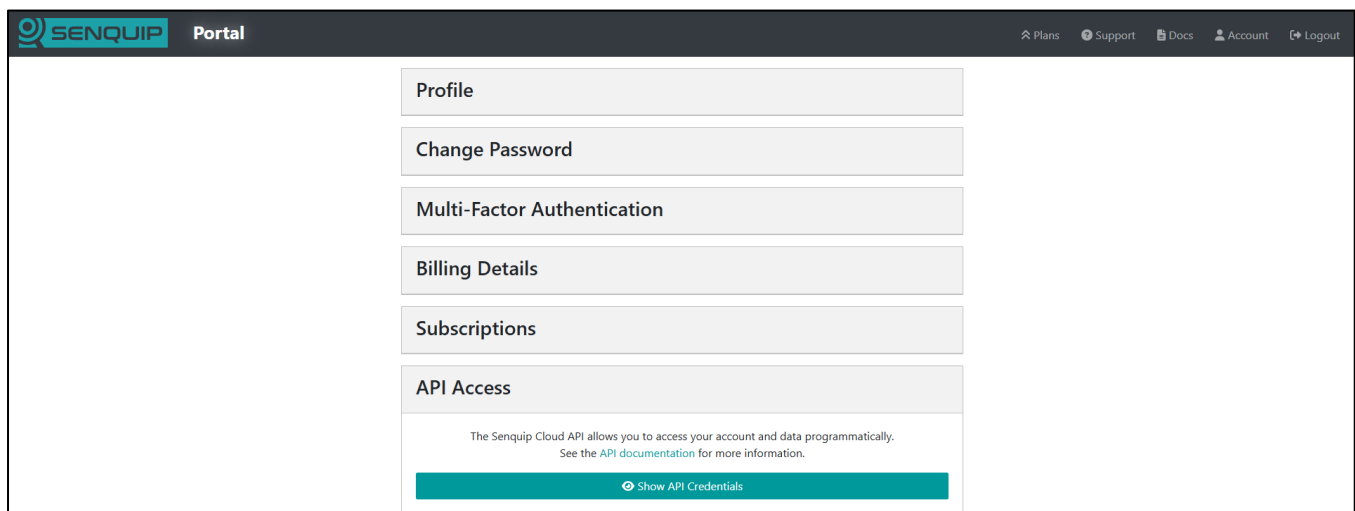


Figure 1 - Generating an API Key

2.2. Retrieving Device Data

The Get Device Data endpoint retrieves data for a given device over a specified time.

<https://api.senquip.com/cloud-api-v2/device/DEVICEID/data>

Parameters

The time range of the data can be customised as required using optional query parameters. If no query parameters are provided, it returns the latest data point.

- **relative:** Request data relative to the current time. Examples: 1m = last one minute, 3d = last three days, 5w = last five weeks, 2M = last two months.

"<https://api.senquip.com/cloud-api-v2/device/DEVICEID/data?relative=5m>"

- **dataPoints:** Maximum number of data points to return.

"<https://api.senquip.com/cloud-api-v2/device/DEVICEID/data?dataPoints=10>"

- **fromTime / toTime:** Start and end of the time range, specified as Unix timestamps (seconds).

"<https://api.senquip.com/cloud-api-v2/device/data/DEVICEID?fromTime=1724901311.8&toTime=1724901551.8>"

Document Number APN0045	Revision 1.0	Prepared By NGB	Approved By NB
Title Senquip API Data Access with Python			Page 3 of 8

- **pageStartTime**: Used for pagination. If the response includes a `lastEvaluatedTime`, pass this value into `pageStartTime` for the next request to continue retrieving data.
- **dataKeys**: Restrict results to specific measurement fields. Can be repeated for multiple keys.

"https://api.senquip.com/cloud-api-v2/device/DEVICEID/data?relative=10m&dataKeys=vin&dataKeys=ambient"

Response

Requests can take up to 20 seconds when retrieving data over a large time range. Set your request timeout accordingly!

- **data**: An array of data points collected from the device.
- **lastEvaluatedTime**: A timestamp (Unix timestamps in seconds) used for pagination. If present, it should be provided in the next request to retrieve the next page of results. If null, there is no more data to paginate.
- **deviceid**: The id of the device the data belongs to.
- **dataCount**: The number of data points returned.
- **message**: Informational status message.

There is an internal limit of how much data can be returned per request. If a non-null value for *lastEvaluatedTime* is returned, then there is more data available for the given parameters. In the example below, all the requested data has been returned as the *lastEvaluatedTime* is showing as null.

The *lastEvaluatedTime* can be passed into a new query as the *pageStartTime* to continue the request.

```
{
  "data": [
    {
      "ts": 1729462721.7,
      "ambient": 24.65,
      "cp3": "MCswKzAuMDAwKzArMCswLjEzKzE3Ny40KzAuMTYrMTkuNisxLjU5KzAuMTMrMC4wMQ0K",
      "ttl": 1792534721,
      "cp9": 0.13,
      "cp8": 0,
      "current4": 0.15,
      "state": 0,
      "wifi_ip": "192.168.1.216",
      "deviceid": "ZSBDH91FF",
      "vreg": 9.06,
      "vsys": 4.16,
      "cycle": 3591,
      "aws_ts": 1729462721,
      "cp13": 1.59,
      "wifi_rssi": -40,
      "light": 1,
      "vin": 12.09,
      "analog4": 5.159
    }
  ],
  "lastEvaluatedTime": null,
  "deviceid": "ZSBDH91FF",
  "dataCount": 1,
  "message": "OK"
}
```

Figure 2 - Example Response

Document Number APN0045	Revision 1.0	Prepared By NGB	Approved By NB
Title Senquip API Data Access with Python			Page 4 of 8

3. Example Python Scripts

This application note provides two example Python scripts to demonstrate how to retrieve data from the Senquip Cloud API V2 and store it locally. They are intended as quick start references and can be adapted to your own projects.

The example scripts are intentionally lightweight and use only the requests package in addition to standard Python libraries. This keeps them easy to run without complex setup.

Both scripts handle pagination automatically using `lastEvaluatedTime`. Large datasets will be split across multiple requests until all points are retrieved.

Appendix 1 – JSON Output

Saves all retrieved datapoints into a single JSON file (`data.json`). This format is ideal for developers who plan to feed the data into applications, dashboards, or analytics pipelines.

Returns all available fields, but the file may be large if the requested time range is long.

Appendix 2 – CSV Output

Saves selected datapoint fields into a CSV file (`data.csv`). This is convenient for engineers and analysts who want to review or graph data in Excel, Google Sheets, or other spreadsheet tools.

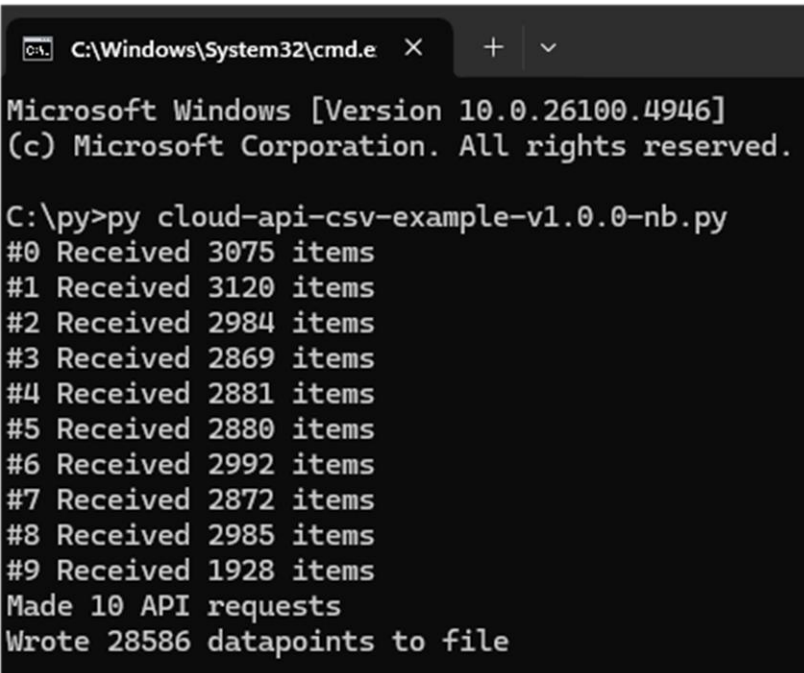
Limits file size by only writes the keys specified in the `csv_keys` list.

3.1. How to Run the Scripts

To demonstrate how to access data through the Senquip Cloud API V2, this application note includes two simple Python scripts. They are lightweight, easy to run, and serve as starting points for your own integrations.

1. Install Python 3 (if not already installed).
2. Install the requests package using pip: `>>pip install requests`
3. Edit the script:
 - a. Replace `api_key` with your own API key generated from the Senquip Portal.
 - b. Replace `deviceid` with the identifier of your Senquip device.
 - c. Adjust `start_time` and `end_time` (Unix timestamps in seconds) or use relative parameters as needed.
 - d. CSV example only - edit the `csv_keys` list to specify which datapoint fields should be written to the CSV file.
4. Run the script: `>>python scriptname.py`

The script will make one or more API calls, display progress in the console, and write the data to a file in the same folder.



```
C:\Windows\System32\cmd.e X + v
Microsoft Windows [Version 10.0.26100.4946]
(c) Microsoft Corporation. All rights reserved.

C:\py>py cloud-api-csv-example-v1.0.0-nb.py
#0 Received 3075 items
#1 Received 3120 items
#2 Received 2984 items
#3 Received 2869 items
#4 Received 2881 items
#5 Received 2880 items
#6 Received 2992 items
#7 Received 2872 items
#8 Received 2985 items
#9 Received 1928 items
Made 10 API requests
Wrote 28586 datapoints to file
```

Figure 3 - Running the Python Script – Multiple Requests were Performed

4. Conclusion

The example scripts in this application note are designed to help you get started quickly with the [Senquip Cloud API V2](#). Whether exporting data as JSON for use in applications, or as CSV for analysis in spreadsheets, the scripts show the essentials of authenticating, retrieving, and storing device measurements.

They are intended as a foundation — you can adapt them to build automated data pipelines, monitoring dashboards, or custom reporting tools tailored to your requirements.

For more advanced usage, including device management and provisioning, refer to the full Senquip API V2 documentation available through the Senquip Portal.

Appendix 1: Extracting Data as JSON

```
# =====
# Example python script to request data from the nominated
# deviceid and save it to a file in JSON format
#
# v2.0.0, Copyright Senquip 2025
# =====

import requests, json
import time

# Replace this value with your own credentials:
api_key = "XXXXXXXXXX"

# Device to request data from
deviceid = 'XXXXXXXXX'

start_time = 1756230395
end_time = int(time.time())

# ===== Function: Request data for a device =====
def getData(deviceid, lastEvaluatedTime = None):
    url = "https://api.senquip.com/cloud-api-v2/device/" + deviceid + "/data"

    headers = {
        "x-api-key": api_key,
        "Content-Type": "application/json",
        "Accept-Encoding": "gzip, deflate"
    }

    params = {
        'fromTime': start_time,
        'toTime': end_time,
        'pageStartTime': lastEvaluatedTime
    }

    response = requests.request("GET", url, headers=headers, data={}, params=params)
    return json.loads(response.text)

## Get data for the device
request_count = 0
data_count = 0
lastEvaluatedTime = None
with open('data.json', 'w') as file:
    file.write('[')
    while ((lastEvaluatedTime is not None) or (data_count == 0)):
        response = getData(deviceid, lastEvaluatedTime)
        print("#%d Received %d items"%(request_count, len(response['data'])))
        for i in response['data']:
            if data_count != 0: file.write(",")
            file.write(str(i).replace(" ", "").replace("'", "\'"))
            file.flush()
            data_count = data_count + 1

        if "lastEvaluatedTime" in response:
            lastEvaluatedTime = response['lastEvaluatedTime']
        else:
            lastEvaluatedTime = None
            request_count = request_count + 1
    file.write(']')

print("Made %d API requests"%request_count)
print("Wrote %d datapoints to file"%data_count)
```

Appendix 2: Extracting Data as CSV

```
# =====
# Example python script to request data from the nominated
# deviceid and save it to a file in CSV format
#
# v1.0.0, Copyright Senquip 2025
# =====

import requests, json, base64
import time

# Replace this value with your own credentials:
api_key = "XXXXXXXX"

# Device to request data from
deviceid = 'XXXXXXX'

# Keys to save as CSV
csv_keys = [
    "ts",
    "ambient",
    "cp1",
    "cp2",
]

start_time = 1756734370
end_time = int(time.time())

# ===== Function: Request data for a device =====
def getData(deviceid, lastEvaluatedTime = None):
    url = "https://api.senquip.com/cloud-api-v2/device/" + deviceid + "/data"

    headers = {
        "x-api-key": api_key,
        "Content-Type": "application/json",
        "Accept-Encoding": "gzip, deflate"
    }

    params = {
        'fromTime': start_time,
        'toTime': end_time,
        'pageStartTime': lastEvaluatedTime
    }

    response = requests.request("GET", url, headers=headers, data={}, params=params)
    return json.loads(response.text)

## Get data for the device
request_count = 0
data_count = 0
lastEvaluatedTime = None
with open('data.csv', 'w') as file:
    # Write CSV header
    file.write(",".join(csv_keys))
    file.write("\n")
    while ((lastEvaluatedTime is not None) or (data_count == 0)):
        response = getData(deviceid, lastEvaluatedTime)
        print("#%d Received %d items"%(request_count, len(response['data'])))
        for i in response['data']:
            if data_count != 0: file.write("\n")
            # Write CSV data, blank if not present, decode base64 strings
            value = []
            for key in csv_keys:
```

Document Number
APN0045

Revision
1.0

Prepared By
NGB

Approved By
NB

Title
Senquip API Data Access with Python

Page
8 of 8

```
v = i.get(key, "")
# Try to decode base64 if value is a string and looks like base64
if isinstance(v, str):
    try:
        # Try to decode, if not base64, just use original
        decoded = base64.b64decode(v, validate=True)
        # If decoded is printable, use it as string
        try:
            decoded_str = decoded.decode('utf-8')
            v = decoded_str
        except Exception:
            v = v # keep as original if not utf-8
        print("except: ", v)
    except Exception:
        pass # not base64, keep as is
value.append(str(v))

# put quotes around each value
file.write(",".join(['"%s"' % x.replace('"', '"') for x in value]))
file.flush()
data_count = data_count + 1;

if "lastEvaluatedTime" in response:
    lastEvaluatedTime = response['lastEvaluatedTime']
else:
    lastEvaluatedTime = None
request_count = request_count + 1

print("Made %d API requests"%request_count)
print("Wrote %d datapoints to file"%data_count)
```