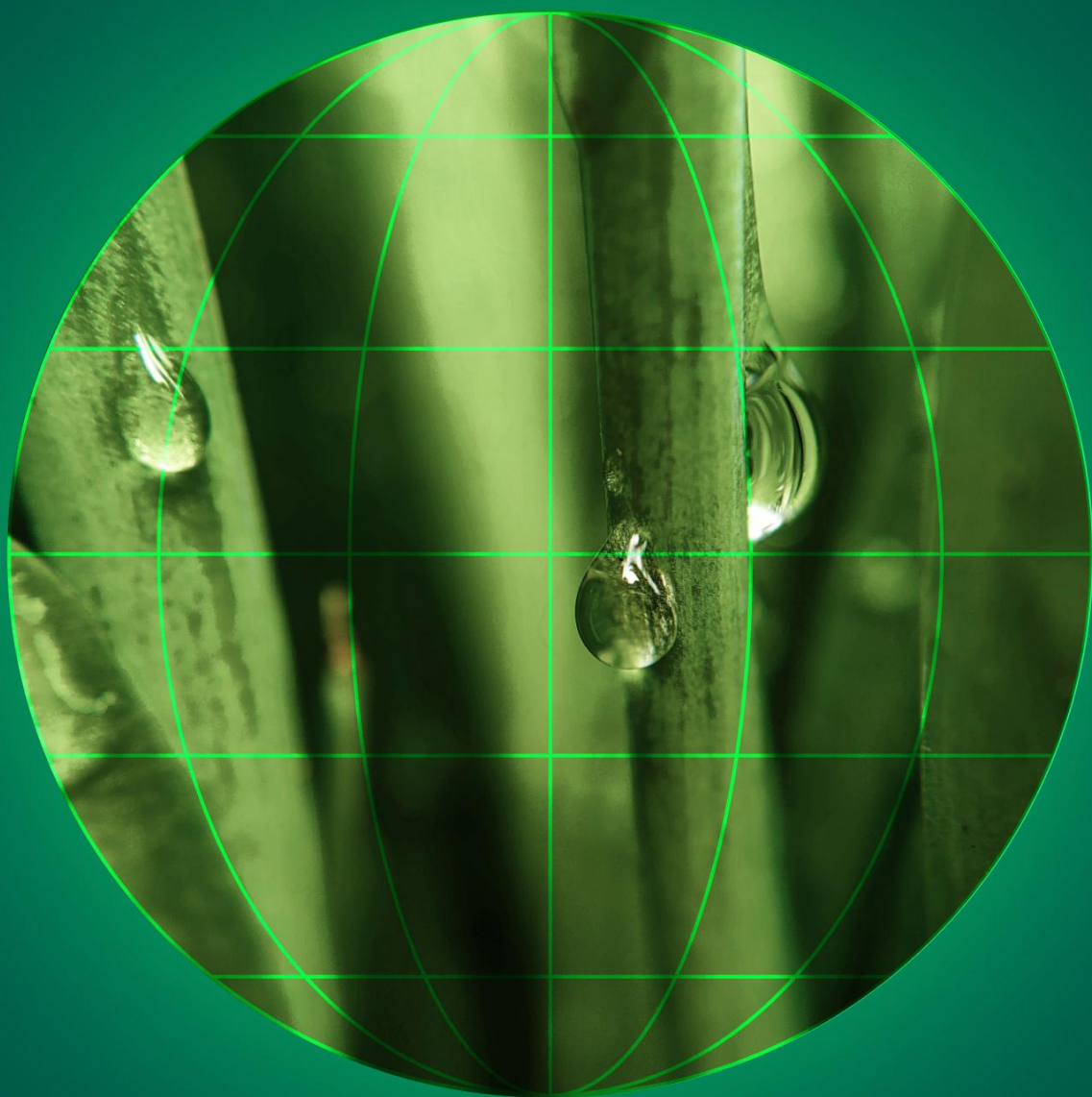


IrriWatch

API Documentation



Version 1.12

August 2026

All rights reserved. The document may not be reproduced or distributed, in whole or in part, without the prior written permission of Hydrosat B.V.

1 Introduction

The IrriWatch API allows companies to programmatically interact with the IrriWatch application.

By using the API, it is possible to create orders for irrigation advice and download data outputs regularly. The API uses standard methods like OAuth2 Client Credentials for authentication and Swagger for the API definition and online manipulation.

To protect the service and ensure consistent performance for all users, API requests are rate-limited. Each IP address is allowed up to 6,000 requests per minute, evaluated over a rolling (sliding) time window. Requests exceeding this limit may be temporarily delayed or throttled.

This document describes at a high level the API provided. For lower-level details please see the Swagger document online: <https://irriwatch.hydrosat.com/api-v1-swagger>

Within the API two different use cases can be identified:

- The order is created for a third party and users of this third party use the IrriWatch portal for viewing the results.
- The order is created for the API customer who will use the generated results for his/her consumption.

IrriWatch is in active development and will be changed over time. Changes will be communicated with API customers via notifications and up to date documentation accessible on the portal.

Attached to this document is the list of all the available pixel and field level outputs of the IrriWatch API.

2 Definition

Within this document, the following definitions are used:

Term	Definition
API Customer	The customer that uses the API
Company	An end customer for whom and order can be created. This can be either end customers or the API customer

User	An end-user who can log in on IrriWatch. A user is uniquely identified by their email address
Order	An order to produce irrigation advice during a period
Order Results	A ZIP file with results
GeoTIFF	A geo-reference tiff file with one or more layers. Generally produced in EPSG:4326/WGS84
Time-Series	A single measure (eg. Average vegetation cover) for a single field, calculated with daily timesteps. Allows e.g. for plotting and shapefile colouring

3 About Orders

An order is created using the API. The order has an active period between the order start date and by default 365 days after or if determined differently with the subscription agreement. Order results will only be produced within a predefined period.

For details on how is determined, when and how order results are produced see the IrriWatch website: <https://irriwatch.hydrosat.com>

As part of the order, a GeoJSON FeatureCollection must be provided for defining the fields and should have EPSG:4326/WGS84 projection.

Each field defined in the GeoJSON provided with an order must have the following properties (a.k.a. columns in shapefiles):

- Crop type
- Irrigation type
- Planting date (if different from order start date)
- Soil type (optional, by default SoilGrids global dataset is used)

The types are identified by integers, the definition is available on the IrriWatch website and in the Swagger documentation (see the /reference/ endpoints).

Additionally, each field **must** have a name defined. This name can be at most 20 characters long.

The property keys are case sensitive and **must** be provided in lower case.

Since the introduction of the new IrriWatch portal, it is also possible to manage your fields by Farm. Orders submitted by API are automatically added to the Default Farm.

4 API Endpoint

The endpoints required to use the API are the following:

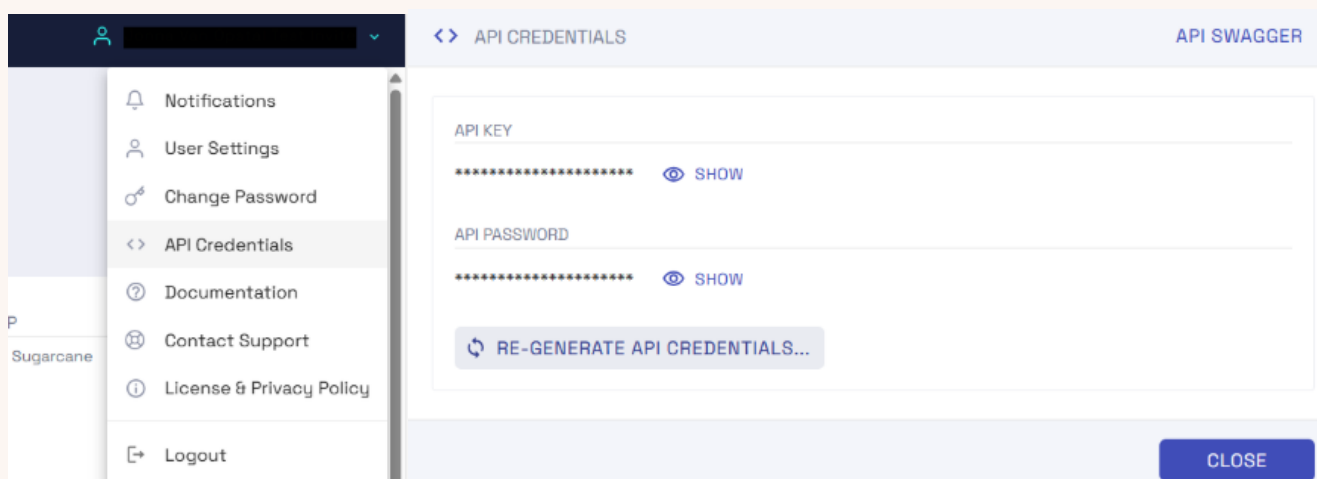
Production endpoint: <https://api.irriwatch.hydrosat.com/api/>

Authentication endpoint: <https://api.irriwatch.hydrosat.com/oauth2/v2/token>

5 Authentication

The API used standardized OAuth2 authentication. The client credentials flow is used.

User API Credentials can be found under the user profile (see below).



After approval for API use, Company API credentials are available on the Company Details page, available under the 'Organization' tab in the IrriWatch portal.

The API key and Password are strictly secret and should be kept safe. If the API key and/or password are lost or compromised, a new key and password needs to be generated.

6 Available API Calls

The available API calls are:

- List companies
- List users for companies
- List, create and complete (end) orders for companies
- List and download order results
- List crop, irrigation, and soil types
- Create meteo records

Please look at the Swagger documentation at <https://irriwatch.hydrosat.com/api-v1-swagger> for calls available.

7 Order Results

As stated in the first paragraph, there are two different use cases for the API:

- For creating orders to be consumed by the API customer.
- For creating orders for end-users that use the IrriWatch portal for viewing data

In the first case, order results consist of:

- ZIP file containing GeoTIFF files per field containing multiple layers

This mode is automatically selected when the company UUID (see the Swagger definition) used for the order is the API customer UUID.

In the second case, the order results consist of:

- Data for use in the frontend, like raster information to be shown on the map and time-series data. This data is **not** downloadable through the API.

This second mode is automatically selected when the company UUID used for the order is NOT the API customer UUID.

Order results are produced in 2 phases: an initial setup phase and a regular run. The initial setup phase may take anywhere between a few hours and a few days.

8 Callbacks

When order results in the form of GeoTIFF ZIP become available a callback to customer systems can be performed.

There are two options to get callbacks: through email and a callback URL. Both can be set on the company profile page at the IrriWatch portal. The callback URL must be accessible from the IrriWatch API server.

The URL used for the callback will be the one entered on the IrriWatch portal and will be appended with `&order=[UUID]` or `?order=[UUID]`.

9 Downloading of Order Results Zips

Data is available in GeoTIFF files, one for each field in an order. The GeoTIFFs will have a fixed order of bands. The available bands and orders depending on the subscription type of the order. Details of what each band means are available on the IrriWatch website.

The GeoTIFF is collected in a ZIP file, that can be downloaded each day. The ZIP is **only** available through the API results endpoint.

10 Order Results From Data Retention

Orders created for customers that have access to the IrriWatch portal, the images shown on the map will be retained for the duration of the order and will be removed afterward. Time series data will be preserved and will be retained for the same period at a minimum.

11 Posting irrigation data

There is the option to post your own recorded irrigation data to the IrriWatch portal. This might be of interest if you want to use your own (locally recorded) data for the models and irrigation advisory forecasts.

The irrigation amounts should be provided with a given date (in YYYY-MM-DD format) and an associated field, noted as fieldID. The fieldID can be found in the field properties (as field UUID).

When posting the irrigation amounts you should take note of the following:

- Indicate the irrigation amounts in mm (millimeters)
- Decimals are indicated as a dot (.) not a comma
- Range that can be provided is 0 – 999

In addition to providing irrigation data, it is also possible to use the same endpoint for providing precipitation data. If only one metric (either irrigation or precipitation data) is provided via the endpoint, then the other metric will not be overwritten but have either null or any previously entered data. A recorded null means that the default IrriWatch calculations will be used for estimating this metric.

Important note: Adding data will only be included in calculations when posted before midnight of the same day. We run our models after midnight to provide data near real-time, the next day.

Further details and an example is provided on the API Swagger:

<https://irriwatch.hydrosat.com/api-v1-swagger>

12 Example Usage Curl

In the next paragraphs, a few examples of the API are given. This is not meant to be complete. Firstly, the generic command-line tool curl is used. Secondly, examples with the Python language are demonstrated (next section).

The example call shown supports the scenario of listing and downloading results for a particular order. It consists of the following steps:

- OAuth2 login
- List accessible companies
- List available orders for a company
- List available results for an order
- Download result ZIP for a particular date

For all calls available look at the Swagger documentation at <https://irriwatch.hydrosat.com/api-v1-swagger>.

12.1 OAuth 2.0 login

The login uses the client credentials flow of OAuth2. In order to login, you need your user API key and password. The call is (replace **highlighted** with relevant info):

```
curl -vvv --request POST --  
url https://api.irriwatch.hydrosat.com/oauth/v2/token --header 'content-type:  
application/x-www-form-urlencoded' --data grant_type=client_credentials --  
data client_id=YOURAPIKEY --data client_secret=YOURPASSWORD
```

When correctly authenticated, an access token will be returned.

Example response:

```
{"token_type":"Bearer","expires_in":604800,"access_token":"YOURTOKEN"}
```

12.2 List accessible companies

Almost all calls need a company UUID. The companies that the API user has access to can be retrieved as follows:

```
curl -s --request GET --url 'https://api.irriwatch.hydrosat.com/api/v1/company'  
-H "Authorization: Bearer YOURTOKEN"
```

The TOKEN is the access token as received by the OAuth2 login call. The result is a JSON array with accessible companies. For each company, the name and UUID are returned.

Example response with 2 companies:

```
[
  {"name":"Your Company Ltd","uuid":"85da35f7-2626-4597-bd2e-
f2e6020667f4"},
  {"name":"Another Company Ltd","uuid":"4add7bf3-5391-4777-86ac-
52c774cbd080"}
]
```

12.3 List available orders for a company

```
curl -s --request GET --
url "https://api.irriwatch.hydrosat.com/api/v1/company/$COMPANY/order" -H
"Authorization: Bearer YOURTOKEN"
```

Example response with 1 order:

```
[
  {"uuid":"165e5e39-b242-4d13-b274-
b09b74d7f5e8","id":12345,"fields":{"type":"FeatureCollection","features":[...],"p
roperties":{"name":"order 123"}"},"state":"canceled","subscription_type":"premi
um"}
]
```

12.4 List available results for an order

```
curl -s --request GET --
url "https://api.irriwatch.hydrosat.com/api/v1/company/$COMPANY/order/$ORD
ERUUID/result" -H "Authorization: Bearer YOURTOKEN"
```

Example response:

```
[
  {"uuid":"12564205-b84a-4ae9-9965-8b7dac983685",
  "name":"20200520.zip","date":"20200520"}
]
```

The UUID is to be used in the download result call.

12.5 Download result ZIP

Results can be downloaded for a single day using the following URL.

```
curl --request GET --
url "https://api.irriwatch.hydrosat.com/api/v1/company/$COMPANY/order/$ORD
ERUUID/result/$FILEID" -H "Authorization: Bearer YOURTOKEN" --output
results.zip
```

The ZIP file contains a few files:

- One TIF file per field containing layers for a few parameters
- A text file containing the names of the layers in the TIF files
- A JSON file containing field-level data

Note that it is not possible to bulk download for multiple days.

13 Example Usage Python

The Python examples shown supports three different scenarios of using the API:

- Downloading field data
- Downloading GeoTIFF data
- Posting Meteo data

In order to complete these scenarios the following three steps need to be taken to start the scenarios:

13.1 Definition of company specific details

In order to login, you need your Company UUID, API Key (Client ID) and Password (Client Secret).

```
# fill in company uuid
company_uuid = ''

# fill in client id and client secret
client_id = ''
client_secret = ''

# get bearer code
login_details = {'grant_type': 'client_credentials',
                 'client_id': client_id,
                 'client_secret': client_secret
                }
```

13.2 Oauth 2.0 login and access token (Bearer code)

With the above created login_details the access token can be retrieved from the API, which is needed to complete the different scenarios.

```
# import modules
import requests
import json

# start api session
session = requests.Session()

# retrieve response of log in into api
token_response = session.post(
    url = 'https://api.irriwatch.hydrosat.com/oauth/v2/token',
    data = login_details
)

# load text from response token
token_dict = json.loads(token_response.text)

# get bearer code from access_token
bearer = token_dict['access_token']

# define authorization details with bearer code
headers = {'Authorization' : 'Bearer ' + bearer}
```

13.3 List available orders for a company

```
# url to retrieve order uuid of company
url_order_uuid =
'https://api.irriwatch.hydrosat.com/api/v1/company/{company_uuid}/order'
.format(
    company_uuid = company_uuid
)

# get url response of api
order_uuid_response = requests.get(
    url = url_order_uuid,
    headers = headers
)

# list of all orders of company
```

13.4 Get field level data in JSON format

In order to get the field level data from the API, the Order UUID and Date need to be specified.

The Order UUID can be found in the list of orders, shown above as `list_order_uuid`.

The date needs to be defined with the following format of year, month, day written as `20240715` (15 July 2024).

```
# select order_uuid from list of uuid orders
order_uuid = ''

# date to retrieve data for, format: year month day
date = '20240715' # example 15 July 2024

# url to retrieve field data of order for specific date
url_field_data =
'https://api.irriwatch.hydrosat.com/api/v1/company/{company_uuid}/order/{order_uuid}/result/{date}/field_level'.format(
    company_uuid = company_uuid,
    order_uuid = order_uuid,
    date = date
)

# get url response of api
field_data_response = requests.get(url = url_field_data, headers =
headers)

# dict with field level data of specific date
field_level_data = json.loads(field_data_response.content)
```

13.5 Get GeoTIFF data with Result UUID

To download the GeoTIFF data from the API a Result UUID needs to be given. In the following section the list of all Result UUIDs are created for an order in a company.

```
# create url to retrieve all tiff result uuid's to retrieve tiff data
url_result_uuids =
'https://api.irriwatch.hydrosat.com/api/v1/company/{company_uuid}/order/{order_uuid}/result'.format(
    company_uuid = company_uuid,
    order_uuid = order_uuid
)

# get url response of api
result_uuids_response = requests.get(
    url = url_result_uuids,
    headers = headers
)

# dict with field level data of specific date
list_result_uuids = json.loads(result_uuids_response.content)
```

13.6 Download zip file containing GeoTIFF files of Result UUID

From the above created list named list_result_uuids a Result_UUID can be selected to download the zip file containing the tiff files for the selected day.

Firstly, two modules need to be imported to download and open the zip file. The path_to_save variable contains the folder path in which the downloaded zip file will be opened.

```
# import modules for download zip file
import zipfile
import io

# select result uuid to download data for
result_uuid = ''

# url to download zip file of result uuid
url_tiff_data =
'https://api.irriwatch.hydrosat.com/api/v1/company/{company_uuid}/order/{order_uuid}/result/{result_uuid}'.format(
    company_uuid = company_uuid,
    order_uuid = order_uuid,
    result_uuid = result_uuid
)

# get url response of api
tiff_data_response = requests.get(
    url = url_tiff_data,
    headers = headers
)

# open response as zipfile object
zipfile_data = zipfile.ZipFile(io.BytesIO(tiff_data_response.content))

# path to extract zipfile data to
path_to_save = ''

# extract all zipped data to given path
zipfile_data.extractall(path_to_save)
```

13.7 Download zip file containing GeoTIFF files by date

Downloading the zip file from above can also be achieved by querying with a specific date. Follow similar steps as above but use a different end point.

```
# import modules for download zip file
import zipfile
import io

# select date (YYYY-MM-DD) to download data for
date = ''

# url to download zip file of result uuid
url_tiff_data =
'https://api.irriwatch.hydrosat.com/api/v1/company/{company_uuid}/order/{order_uuid}/result/date/{date}'.format(
    company_uuid = company_uuid,
    order_uuid = order_uuid,
    date = date
)

# get url response of api
tiff_data_response = requests.get(
    url = url_tiff_data,
    headers = headers
)

# open response as zipfile object
zipfile_data = zipfile.ZipFile(io.BytesIO(tiff_data_response.content))

# path to extract zipfile data to
path_to_save = ''

# extract all zipped data to given path
zipfile_data.extractall(path_to_save)
```

13.8 Post Meteo data

In order to post Meteo data to the API, a JSON file needs to be created with required specifics.

The section below shows an example of posting precipitation data on 15 January 2024 on the location given with latitude and longitude values. Note that time zones are ignored, period is interpreted as UTC. This example shows a location in the Netherlands which has an UTC of +1 in January.

The meteo_extendedAPI endpoint supports one extra parameter, soil temperature, compared to the meteo API endpoint. Additionally, the meteo_extended accepts multiple measurements at different heights for the parameters which include height.

```
# post meteo data
url_meteo_post =
'https://api.irriwatch.hydrosat.com/api/v1/company/{company_uuid}/meteo'
.format(
    company_uuid = company_uuid
)

# amount of precipitation (millimetres) , example of total amount of
precipitation on 15 January 2024
meteo_data = json.dumps({
    'period': {
        'start' : '2024-01-15T01:00:00.465Z',
        'end'   : '2024-01-16T00:59:00.465Z'
    },
    'precipitation': {
        'total': 10.0
    },
    'location': {
        'lat': 51.96204,
        'lon': 5.65325
    }
})

# post meteo data to api
post_precipitation_response = requests.post(url = url_meteo_post,
                                           data = meteo_data,
                                           headers = headers
                                           )

# show details of posted meteo data into the api
print(post_precipitation_response.text)

# expected output
{"period":{"start":"2024-01-15T02:00:00+00:00","end":"2024-01-16T01:59:00+00:00"},"precipitation":{"total":10.0},"location":{"lat":51.
```


API Output of IrriWatch API

Pixel Level output expressed in GeoTIFF files

Output contains one GeoTIFF file per polygon per order per day containing 13 bands

Band no.	TIFF parameter	Parameter explanation	Unit	Spatial resolution	Status
1	Irrigation map	status of irrigation	[1] = urgent irrigation [2] = irrigation [3] = no irrigation [4] = over-irrigation [5] = sparse vegetation	10m	Outdated (replaced by moisture status)
2	Vegetation cover	fraction of soil covered by vegetation	%	10m	Active
3	Soil Moisture Root Zone	average soil moisture in the root zone	m ³ /m ³	10m	Active
4	Leaf Nitrogen	leaf nitrogen % of accumulated actual dry matter production	%	20m	Active
5	NDVI	Normalized difference vegetation index	-	10m	Active
6	Actual Evapotranspiration	actual evapotranspiration	mm/day	10m	Active
7	Actual Crop Production Cumulative	actual dry matter production accumulated from planting date	kg/ha	10m	Active
8	Actual Crop Production	actual dry matter production daily	kg/ha/day	10m	Active
9	Soil Water Potential	averaged soil water potential in the root zone	cm or hPa	10m	Active
10	Moisture Status	status of moisture	[1] = stressed [2] = limited [3] = adequate [4] = wet [5] = sparse vegetation	10m	Active
11	Variable Rate Irrigation Map	irrigation needed considering desirable storage and water requirements	mm	10m	Active
12	Soil Temperature daily	Soil temperature daily average top soil (0 cm)	°C	10m	Active

13	Leaf temperature Satellite Overpass	Leaf temperature at midday (satellite overpass)	°C	10m	Active
----	--	---	----	-----	--------

Field Level output (average values per polygon) expressed in JSON file

Output contains one JSON file for all polygons per order per day

No.	JSON parameter	Parameter explanation	Unit
1	name	polygon name	-
2	date	date	year-month-day
3	actual_crop_production	actual dry matter production	kg/ha/day
4	actual_evapotranspiration	actual evapotranspiration	mm/day
5	actual_evapotranspiration_applied_water	actual evapotranspiration due to applied water	mm/day
6	actual_evapotranspiration_applied_water_cumulative	actual evapotranspiration due to applied water accumulated from planting date	mm
7	actual_evapotranspiration_applied_water_raw	as above (#5) but without post-processing	mm
8	actual_evapotranspiration_applied_water_cumulative_raw	as above (#6) but without post-processing	mm
9	actual_evapotranspiration_precipitation	actual evapotranspiration due to precipitation	mm
10	actual_evapotranspiration_precipitation_cumulative	actual evapotranspiration due to precipitation accumulated from planting date	mm
11	actual_evapotranspiration_cumulative	actual evapotranspiration accumulated from planting date	mm
12	actual_transpiration	actual transpiration	mm/day
13	actual_transpiration_cumulative	actual transpiration accumulated from planting date	mm
14	advised_applied_water	advised irrigation water tomorrow	mm
15	air_temperature_min_24	minimum air temperature within 24 hours	°C

16	air_temperature_24	air temperature averaged over 24 hours	°C
17	applied_water_avg_10d	applied water averaged over the last 10 days	mm/day
18	applied_water_fcdp0	advised irrigation water day 0 (= date json + 1day)	mm/day
19	applied_water_fcdp1	advised irrigation water day 1 (= date json + 2days)	mm/day
20	applied_water_fcdp2	advised irrigation water day 2 (= date json + 3 days)	mm/day
21	applied_water_fcdp3	advised irrigation water day 3 (= date json + 4 days)	mm/day
22	applied_water_fcdp4	advised irrigation water day 4 (= date json + 5 days)	mm/day
23	applied_water_fcdp5	advised irrigation water day 5 (= date json + 6 days)	mm/day
24	applied_water_fcdp6	advised irrigation water day 6 (= date json + 7 days)	mm/day
25	applied_water_fcdp7	advised irrigation water day 7 (= date json + 8 days)	mm/day
26	area_percentage_over_irrigation	percentage of pixels being over irrigated	%
27	area_percentage_requiring_irrigation	percentage of pixels requiring irrigation	%
28	attainable_crop_production	highest reachable dry matter production in the same field	kg/ha/day
29	attainable_crop_production_cumulative	accumulation from planting date for highest reachable dry matter production	kg/ha
30	clay_fraction	percentage clay in soil texture	%
31	critical_soil_moisture_root_zone	critical soil moisture at which crop stress starts	m ³ /m ³
32	crop_production_cumulative	actual dry matter production accumulated from planting date	kg/ha
33	crop_production_below_cumulative	fraction of dry matter production below ground	kg/ha

34	crop_production_above_cumulative	fraction of dry matter production above ground	kg/ha
35	crop_production_percentile5	lowest dry matter production (5th percentile) in the same field	kg/ha/day
36	crop_production_percentile95	highest dry matter production (95th percentile) in the same field	kg/ha/day
37	desirable_storage	desired soil moisture level (between critical and field capacity) used for the irrigation planner; defined by user input or default	[-]
38	effective_root_depth_value	fraction of effective root depth compared to total root depth	[-]
39	et_fcdp0	actual evapotranspiration forecast day 0 (= date json +1 day)	mm/day
40	et_fcdp1	actual evapotranspiration forecast day 1 (=date json + 2 day)	mm/day
41	et_fcdp2	actual evapotranspiration forecast day 2 (=date json + 3 days)	mm/day
42	et_fcdp3	actual evapotranspiration forecast day 3 (= date json + 4 days)	mm/day
43	et_fcdp4	actual evapotranspiration forecast day 4 (= date json + 5 days)	mm/day
44	et_fcdp5	actual evapotranspiration forecast day 5 (= date json + 6 days)	mm/day
45	et_fcdp6	actual evapotranspiration forecast day 6 (= date json + 7 days)	mm/day
46	et_fcdp7	actual evapotranspiration forecast day 7 (= date json + 8 days)	mm/day
47	field_efficiency_avg_10d	field efficiency averaged over the last 10 days	%
48	forecast_et0_8days_sum	reference evapotranspiration forecast accumulated (day json +1 to 8 days)	mm
49	fullness	fullness of the root zone between field capacity and wilting point	%

50	irrigation_amount_cumulative	applied water accumulated from planting date	mm
51	irrigation_type	irrigation type of the field, defined by user input or default	
52	irrigation_yesno	irrigation advice for tomorrow	yes (1) or no (0)
53	leaf_nitrogen	leaf nitrogen % of accumulated actual dry matter production	%
54	leaf_temperature_inst	leaf temperature at midday (satellite overpass)	°C
55	n_leaf_production	accumulation of nitrogen from planting date	kg/ha
56	leaf_water_potential	leaf water potential	bar
57	ls_day	landsat day	yes (1) or no (0)
58	moisture_content_cane	moisture content of sugarcane	%
59	max_applied_water	dimension of irrigation system defined by user input or default indicating maximum volume of water that can be applied	[mm]
60	min_applied_water	defined by user input or default indicating minimum volume of water that can be applied to trigger irrigation advice	[mm]
61	min_irrigation_efficiency	dimension of irrigation system defined by user input or default indicating amount of water used by crop (versus runoff/percolation)	[-]
62	min_irrigation_rotation	dimension of irrigation system defined by user input or default indicating the minimum days between irrigation events	days
63	ndvi	normalized difference vegetation index	-
64	net_carbon_supply_in_soil	net carbon supply in soil	kg/C/ha/day
65	organic_matter	organic matter in the soil	%

66	penmann_monteith_crop_coefficient_kc_avg_10d	crop coefficient based for penman monteith equation averaged over the last 10 days	-
67	penmann_monteith_reference_et0	reference evapotranspiration	mm/day
68	potential_evapotranspiration	potential evapotranspiration	mm/day
69	potential_transpiration	potential transpiration	mm/day
70	precipitation_avg_10d	precipitation averaged over the last 10 days	mm/day
71	precipitation_cumulative	precipitation accumulated from planting date	mm
72	precipitation_fcdp0	precipitation forecast day 0 (= date json + 1 day)	mm/day
73	precipitation_fcdp1	precipitation forecast day 1 (= date json + 2 days)	mm/day
74	precipitation_fcdp2	precipitation forecast day 2 (= date json + 3 days)	mm/day
75	precipitation_fcdp3	precipitation forecast day 3 (= date json + 4 days)	mm/day
76	precipitation_fcdp4	precipitation forecast day 4 (= date json + 5 days)	mm/day
77	precipitation_fcdp5	precipitation forecast day 5 (= date json + 6 days)	mm/day
78	precipitation_fcdp6	precipitation forecast day 6 (= date json + 7 days)	mm/day
79	precipitation_fcdp7	precipitation forecast day 7 (= date json + 8 days)	mm/day
80	production_gap	difference between attainable production and actual dry matter production	kg/ha/day
81	quality_per	general quality percentage of the data based on satellite imagery availability	%
82	relative_et_deficit_daily	relative ET deficit (ET deficit / potential ET)	-
83	relative_humidity_24	air humidity averaged over 24 hours	%
84	root_depth	effective root depth	mm

85	s2_day	sentinel-2 day	yes (1) or no (0)
86	sand_fraction	percentage sand in soil texture	%
87	soil_moisture_holding_capacity	amount of water that the soil can hold for crop use	mm/m
88	soil_moisture_percentile5	lowest soil moisture (5% percentile) in the same field	m ³ /m ³
89	soil_moisture_percentile95	highest soil moisture (95% percentile) in the same field	m ³ /m ³
90	soil_moisture_root_zone	average soil moisture in the root zone	m ³ /m ³
91	soil_moisture_root_zone_irri_fcdp0	soil moisture forecast if irrigation is applied day 0 (= date json + 1 day)	m ³ /m ³
92	soil_moisture_root_zone_irri_fcdp1	soil moisture forecast if irrigation is applied day 1 (= date json + 2 days)	m ³ /m ³
93	soil_moisture_root_zone_irri_fcdp2	soil moisture forecast if irrigation is applied day 2 (= date json + 3 days)	m ³ /m ³
94	soil_moisture_root_zone_irri_fcdp3	soil moisture forecast if irrigation is applied day 3 (= date json + 4 days)	m ³ /m ³
95	soil_moisture_root_zone_irri_fcdp4	soil moisture forecast if irrigation is applied day 4 (= date json + 5 days)	m ³ /m ³
96	soil_moisture_root_zone_irri_fcdp5	soil moisture forecast if irrigation is applied day 5 (= date json + 6 days)	m ³ /m ³
97	soil_moisture_root_zone_irri_fcdp6	soil moisture forecast if irrigation is applied day 6 (= date json + 7 days)	m ³ /m ³
98	soil_moisture_root_zone_irri_fcdp7	soil moisture forecast if irrigation is applied day 7 (= date json + 8 days)	m ³ /m ³
99	soil_moisture_root_zone_no_irri_fcdp0	soil moisture forecast if irrigation is not applied day 0 (= date json + 1 day)	m ³ /m ³
100	soil_moisture_root_zone_no_irri_fcdp1	soil moisture forecast if irrigation is not applied day 1 (= date json + 2 days)	m ³ /m ³
101	soil_moisture_root_zone_no_irri_fcdp2	soil moisture forecast if irrigation is not applied day 2 (= date json + 3 days)	m ³ /m ³

102	soil_moisture_root_zone_no_irri_fcdp3	soil moisture forecast if irrigation is not applied day 3 (= date json + 4 days)	m ³ /m ³
103	soil_moisture_root_zone_no_irri_fcdp4	soil moisture forecast if irrigation is not applied day 4 (= date json + 5 days)	m ³ /m ³
104	soil_moisture_root_zone_no_irri_fcdp5	soil moisture forecast if irrigation is not applied day 5 (= date json + 6 days)	m ³ /m ³
105	soil_moisture_root_zone_no_irri_fcdp6	soil moisture forecast if irrigation is not applied day 6 (= date json + 7 days)	m ³ /m ³
106	soil_moisture_root_zone_no_irri_fcdp7	soil moisture forecast if irrigation is not applied day 7 (= date json + 8 days)	m ³ /m ³
107	soil_temperature	soil temperature	°C
108	soil_type	soil type of the field, defined by user input or default soil map	
109	soil_water_potential_critical	critical soil water potential at which crop stress starts	cm or hPa
110	soil_water_potential_fc	field capacity level referring to soil water potential	cm or hPa
111	soil_water_potential_percentile5	lowest soil water potential (5th percentile) in the same field	cm or hPa
112	soil_water_potential_percentile95	highest soil water potential (95th percentile) in the same field	cm or hPa
113	soil_water_potential_root_zone	averaged soil water potential in the root zone	cm or hPa
114	solar_radiation_cumulative	solar radiation accumulated from planting date	W/m ²
115	stomatal_conductance	measure of the degree of stomatal opening	mm/s
116	stress_type	drought sensitivity of the crop variety, defined by user input or default	
117	sugar_content_cane	sugar content of sugarcane	%

118	thermal_degrees_days_cumulative	thermal degree units accumulated from planting date	°C
119	thermal_update	update from thermal satellite measurement	yes (1) or no (0)
120	theta_fc_sub	field capacity level referring to soil moisture	m ³ /m ³
121	theta_sat_sub	saturation level referring to soil moisture	m ³ /m ³
122	theta_wp_sub	wilting point level referring to soil moisture	m ³ /m ³
123	time_until_next_irrigation	number of days until next advised irrigation	days
124	uniformity_soil_moisture	uniformity of soil moisture in the same field (standard deviation / mean)	-
125	uniformity_vegetation_cover_stdmean	uniformity of vegetation cover in the same field (standard deviation / mean)	-
126	utc_ls_datetime	latest Landsat update acquisition date and time	yyyy-mm-dd hh:mm
127	utc_s2_datetime	latest Sentinel-2 update acquisition date and time	yyyy-mm-dd hh:mm
128	vegetation_cover	fraction of soil covered by vegetation	%
129	vegetation_cover_percentile5	minimum fraction of soil covered by vegetation (5th percentile)	%
130	vegetation_cover_percentile95	maximum fraction of soil covered by vegetation (95th percentile)	%
131	visible_update	update from optical satellite measurement	yes (1) or no (0)
132	water_productivity_avg_10d	water productivity (actual dry matter production / actual evapotranspiration) averaged over the last 10 days	kg/m ³ /day
133	water_unlimited_crop_production	fictive actual dry matter production if crop was not under stress	kg/ha/day
134	water_unlimited_crop_production_cumulative	crop production if crop was not under water stress accumulated from planting date	kg/ha

