fao-gismgr project

GISMGR 2.0

API Reference

Prepared by:

NAME	DIVISION	DATE
Davide Battista	CSI	07 December 2022

Revised by:

NAME	DIVISION	DATE

Revision history:

VERSION	ISSUING DATE	DESCRIPTION	REVISED BY
0	07 December 2022	First Release	Davide Battista

Attachments:

TITLE	FILE NAME	VERSION

Acronyms:

API	-	Application Programming Interface
COG	-	Cloud Optimized GeoTIFF
CRS	-	Coordinate Reference System
EPSG	-	European Petroleum Survey Group
FAO	-	Food and Agriculture Organization of the United Nations
GIS	-	Geographic Information System
JSON	-	JavaScript Object Notation
JWT	-	JSON Web Token
PNG	-	Portable Network Graphics
ReST	-	Representational State Transfer
SLD	-	Styled Layer Descriptor
SRS	-	Spatial Reference System
UI	-	User Interface
URL	-	Uniform Resource Locator

Table Of Contents:

1. INTRODUCTION AND OVERVIEW

1.1. Goal

The goal of this document is to provide the reference of the ReST API exposed by the FAO GISMGR 2.0 System. It's a concise reference manual containing all the information required to work with the API, with details about the functions, classes, return types, arguments and more, supported by tutorials and examples.

1.2. About FAO GISMGR 2.0

GISMGR 2.0 is a FAO project to import, manage, share and publish GIS raster data through ReST APIs or Web User Interface. Data owners can use a shared working environment (**`workspace`**) to manage various resources and work with them as a cohesive unit promoting collaboration between multiple businesses or individuals.

- **`users` and `roles`**
  grant access and privileges to users with predefined roles:
  - Admin
  - Editor
  - Data Manager
  - Data Uploader
  - Data Viewer.

- **Cloud Storage buckets**
  `upload` and **`data`** buckets are provided to ingest data, transform and share the results in COG format.

- **`styles`**
  create, import, update and export raster styles and their legends using SLD, JSON or PNG formats.

- **`maps` (single raster layers)**
  ingest, transform, preview and share single raster data and their metadata. Handle different input formats, continuous or categorized rasters, and convert them to COG format, sharing their metadata as JSON files.

1.3. Service Endpoint

A service endpoint is a base URL that specifies the network address of an API service. One service might have multiple service endpoints. This service has the following service endpoint and all URIs below are relative to this service endpoint:

```
https://data.apps.fao.org/gismgr/api/v2
```

2. Authorizing Requests

Some requests your application sends to the FAO GISMGR API must include an authorization token (**`ID token`**). The token identifies a user account. The user account must be previously granted with permissions by the workspace's administrators.

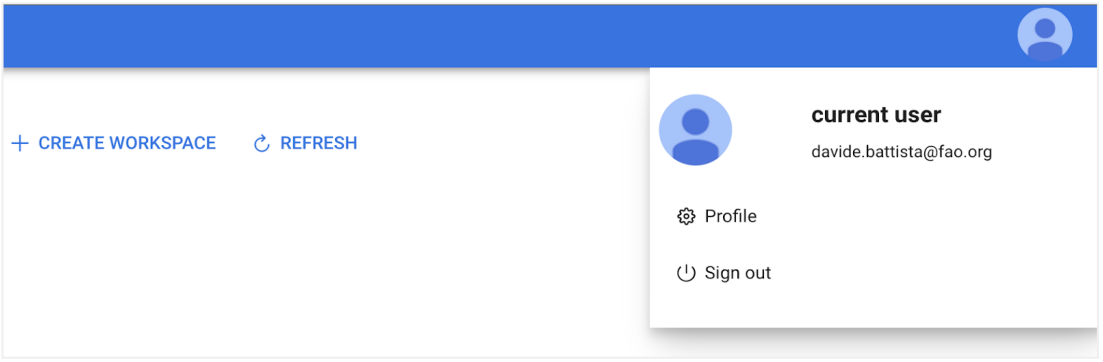
Your client application must send the ID token in the `Authorization` header when making requests for protected resources:

```
Authorization: Bearer <ID_TOKEN>
```

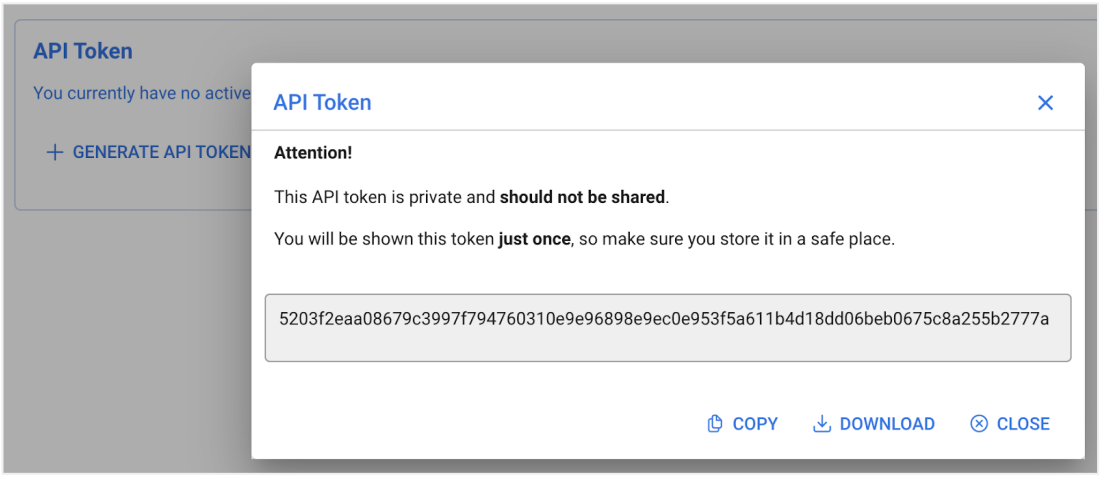
2.1. API Key

In order to retrieve an ID token programmatically, you need to generate the user account's API Key:

- Go to the `GISMGR Web UI` and sign-in.
- In the upper-right corner click on the account icon, then select **`profile`**:

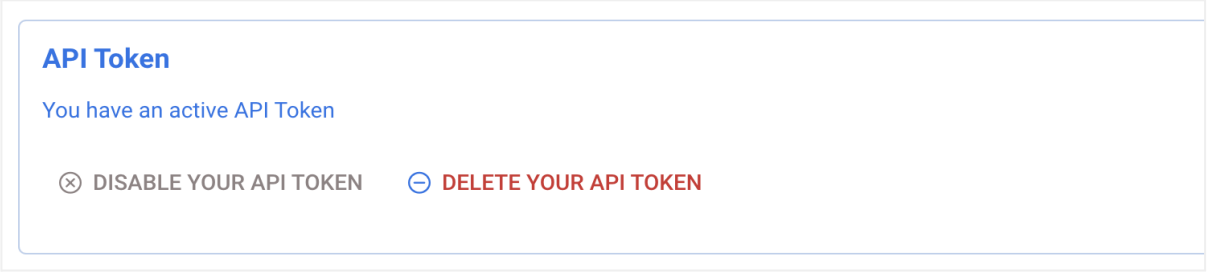


- a new page is displayed with the user account details; hence, in the API Key section, click **`GENERATE API KEY`**:



- the user account API Key will be generated (80 alphanumeric characters string). This will be shown just once (at this time), so it must be copied and stored in a safe place and, above all, should not be shared with anyone.

After an API Key has been generated, it can be temporarily disabled, re-enabled or be definitively deleted (you will be able to re-generate a new API key):



2.2. ID Token

Once you have a valid and enabled API Key, you can request for an ID token.

HTTP Request:

```
POST /catalog/identity/accounts:signInWithApiKey
```

Request Headers:

```
          Accept: application/json
  Content-Length: 0
X-GISMGR-API-KEY: <API_KEY>
```

Response Body:

If successful, the response body contains (among other fields) an instance of:

```
{
    “idToken”: string,
    “refreshToken”: string,
    “expiresIn”: string
}
```

FIELD	DESCRIPTION
idToken	string: an auth ID token
refreshToken	string: a auth refresh token Can be used to refresh if the ID token has been expired or it is going to expire
expiresIn	string: the number of seconds in which the ID token expires.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# enabled user account API Key
API_KEY=5ea2f6941e8d02566c38d4cb2115f38...

# authenticate
curl -X "POST" -H "X-GISMGR-API-KEY: ${API_KEY}" -H “Accept: application/json” -H
"Content-Length: 0" "${BASE_URL}/catalog/identity/accounts:signInWithApiKey"

# RESPONSE:
{
  "requestId": "59134e8f-3b66-4a25-bdfa-0b9f237c634b",
  "timestamp": 1669300855538,
  "runtime": 990,
  "status": 200,
  "message": "OK",
  "response": {
    "idToken": "eyJhbGciOiJSUzI1NiIsImtpZCI6ImE5NmF...",
    "refreshToken": "AOkPPWTOZsM3oTfx2Yf728fmp3UjT8dAsiNmCm1Xd...",
    "expiresIn": "3600"
  }
}
```

2.3. Refresh Token

If a previously retrieved ID token has been expired, or it is going to expire, you can refresh it by issuing the `signInWithRefreshToken` request:

HTTP Request:

```
POST /catalog/identity/accounts:signInWithRefreshToken
```

Request Headers:

```
      Accept: application/json
Content-Type: application/json
```

Request Body:

```
{
  “refreshToken”: string,
}
```

FIELD	DESCRIPTION
refreshToken	string: a previously obtained auth refresh token

Response Body:

If successful, the response body contains (among other fields) an instance of:

```
{
    “idToken”: string,
    “refreshToken”: string,
    “expiresIn”: string
}
```

FIELD	DESCRIPTION
idToken	string: an new (refreshed) auth ID token
refreshToken	string: the auth refresh token provided in the request or a new one
expiresIn	string: the number of seconds in which the ID token expires.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
REFRESH_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# authenticate
curl -X "POST" -H “Accept: application/json” -H "Content-Type: application/json" -d
"{\"refreshToken\": \"${REFRESH_TOKEN}\"}"
"${BASE_URL}/catalog/identity/accounts:signInWithRefeshToken"

# RESPONSE:
{
  "requestId": "59134e8f-3b66-4a25-bdfa-0b9f237c634b",
  "timestamp": 1669300855538,
  "runtime": 580,
  "status": 200,
  "message": "OK",
  "response": {
    "idToken": "eyJhbGciOiJSUzI1NiIsImtpZCI6ImE5NmF...",
    "refreshToken": "AOkPPWTOZsM3oTfx2Yf728fmp3UjT8dAsiNmCm1Xd...",
    "expiresIn": "3600"
  }
}
```

3. Query Parameters

Unless specified otherwise, for all the requests that return **a resource items list**, the following query parameters apply:

PARAMETER	DESCRIPTION
`view`	**optional** the level of the details of the resources returned by the response: • `overview` - only the main fields. • `details` - all the fields except the hidden ones (default) • `hidden` - all the fields
`size`	**optional** max number of resource items per page (paginated request). min value=1; max value=100; default=20
`page`	**optional** number of the page; default=0 → first page (paginated request)
`offset`	**optional** number of resource items to skip (sliced request)
`limit`	**optional** max number of resource items to return (sliced request) min value=1; max value=100;

`sort`	**optional**
	sort the resource items, in ascending or descending order, according to the field(s) specified. Note that page/size (paginated request) or the limit/offset (sliced request) parameters are applied after the full result-set has been sorted.
	
	syntax:
	`sort=field[:{ASC|DESC}][,field[:{ASC|DESC}]][,...]`
	
	• `field` must not be complex type (arrays or objects)
	• ASC and DESC are case insensitive (default=ASC)
	• no space characters are allowed
	
	Example:
	`sort=caption,code:desc`

By default, the request is paginated (page=0; size=20; limit=null; offset=null).

The `page/size` parameters (paginated request) are mutually exclusive with the `limit/offset` ones (sliced request).

To perform a sliced request both the `limit` and the `offset` parameter must be specified (`page` and `size` must be skipped)

Unless specified otherwise, for all the requests that return **a single resource item**, the following query parameters apply:

PARAMETER	DESCRIPTION
`view`	**`optional`** the level of the details of the resources returned by the response: ● `overview` - only the main fields. ● `details` - all the fields except the hidden ones (default) ● `hidden` - all the fields

4. Workspace

A shared working environment to gather various resources and work with them as a cohesive unit promoting collaboration between multiple businesses or individuals.

`Workspace` JSON representation

```
{
  “code”: string,
  “caption”: string,
  “description”: string,
  “contacts”: [
    contact
  ],
  “tags”: [
     string
  ],
  “additionalInfo”: object
}
```

FIELD	DESCRIPTION
code	`string`: unique identifier of the resource. cannot be null; length must be min=3 and max=32; allowed characters can be capital letters (A-Z), digits (0-9), hyphen (-); cannot start and/or end with an hyphen
caption	`string`: a title or brief explanation of the resource. cannot be null; length must be min=3 and max=128; all characters except line breaks are allowed.
description	`string`: full or long explanation of the resource. can be null or empty;
contacts	`array`: list of `contact` objects (see below); can be null or empty.
tags	`array`: list of `string` objects; can be null or empty; a **`tag`** is a keyword or term assigned to a piece of information to describe it.
additionalInfo	`object`: additional information about the resource. can be null; a freely structured JSON object provided by the data owner to better describe the resource

Contact JSON representation

```
{
  “fullName”: string,
  “jobTitle”: string,
  “department”: string,
  “employer”: string,
  “email”: string,
  “phone”: string,
}
```

FIELD	DESCRIPTION
fullName	string: mandatory; the full name of the person.
jobTitle	string: optional; person's official designation in the company. it indicates the seniority level and roles and responsibilities in the organization.
department	string: optional; division of a large organization such as a government, university, or business, dealing with a specific area of activity.
employer	string: optional; individual or an organization in the government, private, nonprofit, or business sector for which the person works.
email	string: mandatory; person’s valid email address.
phone	string: optional; person’s phone number

4.1. Create

To create a workspace, use the following POST request and include the authorization described in Authorizing Requests.

ATTENTION: only System Administrators can create workspaces

HTTP Request:

```
POST /catalog/workspaces
```

Request Headers:

```
       Accept: application/json
 Content-Type: application/json
Authorization: Bearer <ID_TOKEN>
```

Request Body:

The request body is an instance of `workspace`.

Response Body:

If successful (HTTP response code = **201**), the response body contains a newly created instance of `workspace`.

Example:

prepare a `WORKSPACE.json` file with the following content:

```
{
  "code": "ACME",
  "caption": "A Company Making Everything (ACME)",
  "description": "Outlandish products that fail or backfire catastrophically at the
worst possible times.",
  "contacts": [
    {
      "fullName": "Wile Ethelbert Coyote",
      "email": "wile.e.coyote@looney-tunes.com",
      "jobTitle": "Actor",
      "department": "Cartoons",
      "employer": "Looney Tunes"
    },
    {
      "fullName": "Road Runner",
      "email": "road.runner@looney-tunes.com",
      "jobTitle": "Actor",
      "department": "Cartoons",
      "employer": "Looney Tunes"
    }
  ],
  "additionalInfo": {
    "wiki": "https://en.wikipedia.org/wiki/Acme_Corporation"
  },
  "tags": [
    "TEST"
  ]
}
```

execute the request:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# create a workspace
curl -X POST -H "Content-Type: application/json" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -d @WORKSPACE.json
"${BASE_URL}/catalog/workspaces"

# RESPONSE:
{
  "requestId": "3b4c6ad9-6491-4d4d-b2f0-79fd22cf7339",
  "timestamp": 1669378706139,
  "runtime": 57325,
  "status": 201,
  "message": "Created",
  "response": {
   "code": "ACME",
   "caption": "A Company Making Everything (ACME)",
   "description": "Outlandish products that fail or backfire catastrophically at the
worst possible times.",
   "contacts": [
    {
      "fullName": "Wile Ethelbert Coyote",
      "jobTitle": "Actor",
      "department": "Cartoons",
      "employer": "Looney Tunes",
      "email": "wile.e.coyote@looney-tunes.com",
      "phone": null
    },
    {
      "fullName": "Road Runner",
      "jobTitle": "Actor",
      "department": "Cartoons",
      "employer": "Looney Tunes",
      "email": "road.runner@looney-tunes.com",
      "phone": null
    }
   ],
   "additionalInfo": {
    "wiki": "https://en.wikipedia.org/wiki/Acme_Corporation"
   },
   "tags": [
    "TEST"
   ],
   "links": [
    {
      "rel": "parent",
      "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces"
    },
    {
      "rel": "self",
      "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME"
    },
    {
      "rel": "users",
      "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users"
    },
```

```
    {
      "rel": "styles",
      "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles"
    },
    {
      "rel": "maps",
      "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps"
    }
   ]
  }
}
```


4.2. List

Retrieves a list of `workspace` resources (paginated or sliced).

Authentication/Authorization is not required

HTTP Request:

`GET /catalog/workspaces`

Request Headers:

`Accept: application/json`

Query Parameters:

The default query parameters apply.

Response Body:

If successful (HTTP response code = **200**), the response body contains an array of `workspace` items.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# list (paginated: page=0 - first page; size=20; view=overview)
curl -X GET -H "Accept: application/json"
"${BASE_URL}/catalog/workspaces?view=overview"

# RESPONSE
{
  "requestId": "efc6a247-b194-4beb-99eb-3d4c46d94d2d",
  "timestamp": 1669386848043,
  "runtime": 13,
  "status": 200,
  "message": "OK",
  "response": {
    "size": 20,
    "number": 0,
    "hasPrevious": false,
    "hasNext": false,
    "items": [
      {
        "code": "ACME",
        "caption": "A Company Making Everything (ACME)",
        "tags": [
          "TEST"
        ],
        "links": [
          {
            "rel": "self",
            "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME"
          }
        ]
      },
      {
         . . .
      },
      . . .
    ],
    "links": [
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces?view=overview"
      }
    ]
  }
}
```

4.3. Get

Retrieves a selected `workspace` resource item.

Authentication/Authorization is not required

HTTP Request:

`GET /catalog/workspaces/{w_code}`

Request Headers:

`Accept: application/json`

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace you want to retrieve

Query Parameters:

The default query parameters apply.

Response Body:

If successful (HTTP response code = **200**), the response body contains an instance of a `workspace` resource item.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# get (view=details)
curl -X GET -H "Accept: application/json" "${BASE_URL}/catalog/workspaces/ACME"

# RESPONSE
{
  "requestId": "63f49afe-fcda-459f-858e-99d4c87c51e0",
  "timestamp": 1669631375720,
  "runtime": 16,
  "status": 200,
  "message": "OK",
  "response": {
    "code": "ACME",
    "caption": "A Company Making Everything (ACME)",
    "description": "Outlandish products that fail or backfire catastrophically at
the worst possible times.",
    "contacts": [
      {
        "fullName": "Wile Ethelbert Coyote",
        "jobTitle": "Actor",
        "department": "Cartoons",
        "employer": "Looney Tunes",
        "email": "wile.e.coyote@looney-tunes.com",
```

```
        "phone": null
      },
      {
        "fullName": "Road Runner",
        "jobTitle": "Actor",
        "department": "Cartoons",
        "employer": "Looney Tunes",
        "email": "road.runner@looney-tunes.com",
        "phone": null
      }
    ],
    "additionalInfo": {
      "wiki": "https://en.wikipedia.org/wiki/Acme_Corporation"
    },
    "tags": [
      "TEST"
    ],
    "links": [
      {
        "rel": "parent",
        "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces"
      },
      {
        "rel": "self",
        "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME"
      },
      {
        "rel": "users",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users"
      },
      {
        "rel": "styles",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles"
      },
      {
        "rel": "maps",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps"
      }
    ]
  }
}
```


4.4. Update

To update a workspace, use the following PUT request and include the authorization described in Authorizing Requests.

HTTP Request:

```
PUT /catalog/workspaces/{w_code}
```

Request Headers:

```
       Accept: application/json
 Content-Type: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace you want to update

Request Body:

The request body is an instance of workspace.

Response Body:

If successful (HTTP response code = **200**), the response body contains the updated instance of the selected workspace.

Example:

prepare or update the WORKSPACE.json file containing an instance of workspace resource item. (see Create chapter)

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# update an existing workspace (tags and additionalInfo fields)
curl -X PUT -H "Content-Type: application/json" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -d @WORKSPACE.json
"${BASE_URL}/catalog/workspaces/ACME"

# RESPONSE:
{
  "requestId": "3ac32200-e331-44f0-911e-c9bc9d9b171b",
  "timestamp": 1669632926607,
  "runtime": 334,
  "status": 200,
  "message": "OK",
  "response": {
    "code": "ACME",
    "caption": "A Company Making Everything (ACME)",
    "description": "Outlandish products that fail or backfire catastrophically at
the worst possible times.",
```

```
    "contacts": [
      {
        "fullName": "Wile Ethelbert Coyote",
        "jobTitle": "Actor",
        "department": "Cartoons",
        "employer": "Looney Tunes",
        "email": "wile.e.coyote@looney-tunes.com",
        "phone": null
      },
      {
        "fullName": "Road Runner",
        "jobTitle": "Actor",
        "department": "Cartoons",
        "employer": "Looney Tunes",
        "email": "road.runner@looney-tunes.com",
        "phone": null
      }
    ],
    "additionalInfo": {
      "wiki": [
        "https://en.wikipedia.org/wiki/Acme_Corporation",
        "https://en.wikipedia.org/wiki/Looney_Tunes",
        "https://en.wikipedia.org/wiki/Warner_Bros"
      ]
    },
    "tags": [
      "CARTOON",
      "TEST"
    ],
    "links": [
      {
        "rel": "parent",
        "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces"
      },
      {
        "rel": "self",
        "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME"
      },
      {
        "rel": "users",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users"
      },
      {
        "rel": "styles",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles"
      },
      {
        "rel": "maps",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps"
      }
    ]
  }
}
```

4.5. Delete

Deletes an existing `workspace` resource item. Include the authorization described in Authorizing Requests.

ATTENTION: only System Administrators can delete workspaces

HTTP Request:

```
DELETE /catalog/workspaces/{w_code}
```

Request Headers:

```
       Accept: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace you want to delete

Response Body:

If successful (HTTP response code = **200**), the response body contains the deleted instance of the selected `workspace`.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# delete an existing workspace
curl -X DELETE -H "Accept: application/json" -H "Authorization: Bearer ${ID_TOKEN}"
"${BASE_URL}/catalog/workspaces/ACME"

# RESPONSE:
{
  "requestId": "3ac32200-e331-44f0-911e-c9bc9d9b171b",
  "timestamp": 1669632926607,
  "runtime": 334,
  "status": 200,
  "message": "OK",
```

```
  "response": {
    "code": "ACME",
    "caption": "A Company Making Everything (ACME)",
    "description": "Outlandish products that fail or backfire catastrophically at
the worst possible times.",
    "contacts": [
      {
        "fullName": "Wile Ethelbert Coyote",
        "jobTitle": "Actor",
        "department": "Cartoons",
        "employer": "Looney Tunes",
        "email": "wile.e.coyote@looney-tunes.com",
        "phone": null
      },
      {
        "fullName": "Road Runner",
        "jobTitle": "Actor",
        "department": "Cartoons",
        "employer": "Looney Tunes",
        "email": "road.runner@looney-tunes.com",
        "phone": null
      }
    ],
    "additionalInfo": {
      "wiki": [
        "https://en.wikipedia.org/wiki/Acme_Corporation",
        "https://en.wikipedia.org/wiki/Looney_Tunes",
        "https://en.wikipedia.org/wiki/Warner_Bros"
      ]
    },
    "tags": [
      "CARTOON",
      "TEST"
    ],
    "links": [
      {
        "rel": "parent",
        "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces"
      },
      {
        "rel": "self",
        "href": "http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME"
      },
      {
        "rel": "users",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users"
      },
      {
        "rel": "styles",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles"
      },
      {
        "rel": "maps",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps"
      }
    ]
  }
}
```

5. User

GISMGR manages resource access permissions using the concept of user and roles. A user is identified by a valid and existing email address. A role specifies which operations that user can perform on resources belonging to a specific workspace

USER JSON representation

```
{
  "email": string,
  "role": string
}
```

FIELD	DESCRIPTION
email	string: the email address that identifies the user. cannot be null; must be a valid and existing email address.
role	string: the role assigned to the user within the selected workspace cannot be null; allowed values are: • ADMIN • EDITOR • DATA_MANAGER • DATA_UPLOADER • DATA_VIEWER

5.1. Roles

ROLE	DESCRIPTION
ADMIN	• can perform READ/WRITE operations on all workspace resources except managing ADMIN members. • can list/read/download data stored on the workspace's DATA bucket. • can list/read/download and store data on the workspace's UPLOAD bucket. Only System Administrators can create, update or delete workspace ADMINs.
EDITOR	• can perform READ/WRITE operations on all workspace resources except managing users and changing DATA bucket accessibility. • can list/read/download data stored on the workspace's DATA bucket. • can list/read/download and store data on the workspace's UPLOAD bucket.

DATA_MANAGER	• can list/read/download data stored on the workspace's DATA bucket. • can list/read/download and store data on the workspace's UPLOAD bucket.
DATA_UPLOADER	• can list/read/download and store data on the workspace's UPLOAD bucket.
DATA_VIEWER	• can list/read/download data stored on the workspace's DATA bucket.

5.2. Create

To create a user, use the following POST request and include the authorization described in Authorizing Requests.

HTTP Request:

```
POST /catalog/workspaces/{w_code}/users
```

Request Headers:

```
      Accept: application/json
Content-Type: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the resources belong and on which you want to grant permissions.

Request Body:

The request body is an instance of `user` resource.

Response Body:

If successful (HTTP response code = **201**), the response body contains a newly created instance of `user` resource.

Example:

prepare a USER.json file with the following content:

```
{
  "email": "duffy.duck@looney-tunes.com",
  "role": "EDITOR"
}
```

execute the request:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# create
curl -X POST -H "Content-Type: application/json" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -d @USER.json
"${BASE_URL}/catalog/workspaces/ACME/users"

# RESPONSE
{
  "requestId": "2b818214-25b7-419a-b2ea-90908626e131",
  "timestamp": 1669639522973,
  "runtime": 2448,
  "status": 201,
  "message": "Created",
  "response": {
    "email": "duffy.duck@looney-tunes.com",
    "role": "EDITOR",
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users/duffy.duck@loo
ney-tunes.com"
      }
    ]
  }
}
```

5.3. List

Retrieves a list of `user` resources. The request must include the bearer ID token.

HTTP Request:

`GET /catalog/workspaces/{w_code}/users`

Request Headers:

```
      Accept: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the resources belong and on which you want to grant permissions.

Query Parameters:

PARAMETER	DESCRIPTION
asMap	boolean; optional (default=false) returns the result as an object instead as an array of user resources

Response Body:

If successful (HTTP response code = **200**), the response body contains an array of user resource items. If the asMap=true query parameter has been specified, the response body contains an object where the key/value pairs represent the user email address and its assigned role.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# users list (as object)
curl -X GET -H "Accept: application/json" -H "Authorization: Bearer ${ID_TOKEN}"
"${BASE_URL}/catalog/workspaces/ACME/users?asMap=true"

# RESPONSE:
{
  "requestId": "785108da-d104-4bd8-a048-d28c9af75463",
  "timestamp": 1669641602394,
  "runtime": 10,
  "status": 200,
  "message": "OK",
  "response": {
    "duffy.duck@looney-tunes.com": "EDITOR",
    "bugs.bunny@looney-tunes.com": "ADMIN"
  }
}
```

5.4. Get

Retrieves an existing `user` resource item. The request must include the bearer ID token.

HTTP Request:

```
GET /catalog/workspaces/{w_code}/users/{email}
```

Request Headers:

```
       Accept: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
`w_code`	the unique code of the workspace to which the resources belong and on which you want to grant permissions.
`email`	the email address that identifies the user/account

Response Body:

If successful (HTTP response code = **`200`**), the response body contains an instance of `user` resource.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# get an existing user
curl -X GET -H "Accept: application/json" -H "Authorization: Bearer ${ID_TOKEN}"
"${BASE_URL}/catalog/workspaces/ACME/users/bugs.bunny@looney-tunes.com"

# RESPONSE:
{
  "requestId": "cde0df79-613d-45a9-a46b-ccba478fc31d",
  "timestamp": 1669647351441,
  "runtime": 18,
  "status": 200,
  "message": "OK",
  "response": {
    "email": "bugs.bunny@looney-tunes.com",
```

```
    "role": "ADMIN",
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users/bugs.bunny@loo
ney-tunes.com"
      }
    ]
  }
}
```


5.5. Update

To update an existing user, use the following PUT request and include the authorization described in Authorizing Requests.

HTTP Request:

```
PUT /catalog/workspaces/{w_code}/users/{email}
```

Request Headers:

```
       Accept: application/json
 Content-Type: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the resources belong and on which you want to grant permissions.
email	the email address that identifies the user/account

Request Body:

The request body is an instance of `user` resource.

Response Body:

If successful (HTTP response code = **200**), the response body contains the updated instance of the existing `user` resource.

Example:

prepare a USER.json file with the following content:

```
{
  "email": "duffy.duck@looney-tunes.com",
  "role": "DATA_VIEWER"
}
```

execute the request:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# update
curl -X PUT -H "Content-Type: application/json" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -d @USER.json
"${BASE_URL}/catalog/workspaces/ACME/users/duffy.duck@looney-tunes.com"

# RESPONSE
{
  "requestId": "2b818214-25b7-419a-b2ea-90908626e131",
  "timestamp": 1669639522973,
  "runtime": 2448,
  "status": 200,
  "message": "OK",
  "response": {
    "email": "duffy.duck@looney-tunes.com",
    "role": "DATA_VIEWER",
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/users/duffy.duck@loo
ney-tunes.com"
      }
    ]
  }
}
```

5.6. Delete

To delete an existing user, use the following DELETE request and include the authorization described in Authorizing Requests.

HTTP Request:

```
DELETE /catalog/workspaces/{w_code}/users/{email}
```

Request Headers:

```
       Accept: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the resources belong and on which you want to grant permissions.
email	the email address that identifies the user/account

Response Body:

If successful (HTTP response code = **200**), the response body contains the deleted instance of the existing user resource.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# update
curl -X DELETE -H "Accept: application/json" -H "Authorization: Bearer ${ID_TOKEN}"
"${BASE_URL}/catalog/workspaces/ACME/users/duffy.duck@looney-tunes.com"

# RESPONSE
{
  "requestId": "2b818214-25b7-419a-b2ea-90908626e131",
  "timestamp": 1669639522973,
  "runtime": 2448,
  "status": 200,
  "message": "OK",
  "response": {
    "email": "duffy.duck@looney-tunes.com",
    "role": "DATA_VIEWER"
  }
}
```

6. Bucket

Two Google Cloud Storage buckets are bound to each workspace:

NAME	DESCRIPTION
UPLOAD	used to ingest/import workspace's data into the system. Data can be uploaded manually, using the Google Cloud Platform console, or programmatically using provided or custom tools. All the predefined roles, except the DATA_VIEWER one, are able to list/read/download and upload data.
DATA	used to store the processed final data and the related metadata files. Data can be uploaded only by the system, while list/read/download operations are allowed depending on the accessibility level type set on the bucket (see below).

6.1. DATA bucket accessibility

Who can list/read/download what has been stored on the DATA bucket depends on the accessibility level type set on it:

TYPE	DESCRIPTION
ALL USERS	all objects stored on the bucket are readable to anyone who is on the internet, including authenticated and unauthenticated users (anonymous visitors).
AUTHENTICATED USERS	all objects stored on the bucket are readable to all service accounts and all users on the internet who have authenticated with a Google Account. This includes accounts that aren't connected to a Google Workspace account or Cloud Identity domain, such as personal Gmail accounts. Users who aren't authenticated, such as anonymous visitors, aren't included.
NOT PUBLIC	all objects stored on the bucket are readable to all service accounts and all users on the internet who have authenticated with a Google Account and have been granted with one of the predefined workspace roles except the DATA_UPLOADER one.

6.2. Update

By default the accessibility level type of the DATA bucket is set equal to NOT

PUBLIC. To update/change the accessibility level type, use the following PUT request and include the authorization described in Authorizing Requests.

HTTP Request:

```
PUT /catalog/workspaces/{w_code}/data-access
```

Request Headers:

```
      Accept: application/json
Content-Type: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the DATA bucket belongs.

Request Body:

```
{
  "type": string
}
```

FIELD	DESCRIPTION
type	string: the new DATA bucket accessibility level type. cannot be null; allowed values are: • ALL USERS • AUTHENTICATED USERS • NOT PUBLIC

Response Body:

If successful (HTTP response code = **200**), the response body contains:

```
{
  “workspaceCode”: string,
  “type”: string
}
```

FIELD	DESCRIPTION
workspaceCode	string: the unique code of the workspace to which the DATA bucket belongs.
type	string: the updated DATA bucket accessibility level type.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# update DATA bucket accessibility
curl -X PUT -H "Content-Type: application/json" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -d '{"type": "ALL USERS"}'
"${BASE_URL}/catalog/workspaces/ACME/data-access"

# RESPONSE
{
  "requestId": "6af40a95-fe53-4a61-b003-683f5e84fdc9",
  "timestamp": 1669720711174,
  "runtime": 1713,
  "status": 200,
  "message": "OK",
  "response": {
    "workspaceCode": "ACME",
    "type": "ALL USERS"
  }
}
```

6.3. Get

To retrieve the accessibility level type of a DATA bucket, use the following GET request.

Authentication/Authorization is not required

HTTP Request:

```
GET /catalog/workspaces/{w_code}/data-access
```

Request Headers:

```
Accept: application/json
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the DATA bucket belongs.

Response Body:

If successful (HTTP response code = **200**), the response body contains:

```
{
  "workspaceCode": string,
  "type": string
}
```

FIELD	DESCRIPTION
workspaceCode	string: the unique code of the workspace to which the DATA bucket belongs.
type	string: the DATA bucket accessibility level type.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2
```

```
# update DATA bucket accessibility
curl -X GET -H "Accept: application/json"
"${BASE_URL}/catalog/workspaces/ACME/data-access"

# RESPONSE
{
  "requestId": "6af40a95-fe53-4a61-b003-683f5e84fdc9",
  "timestamp": 1669720711174,
  "runtime": 1713,
  "status": 200,
  "message": "OK",
  "response": {
    "workspaceCode": "ACME",
    "type": "ALL USERS"
  }
}
```

7. Style

Geospatial data has no intrinsic visual component. In order to see data, it must be styled. Styling specifies color, opacity, and other visible attributes used to render data on a map.

Style JSON representation

```
{
  "workspaceCode": string,
  "code": string,
  "caption": string,
  "description": string,
  "colorMap": ColorMap
}
```

FIELD	DESCRIPTION
workspaceCode	string: unique code of the workspace to which the resource belongs. cannot be null; length must be min=3 and max=32; allowed characters can be capital letters (A-Z), digits (0-9), hyphen (-); cannot start and/or end with an hyphen
code	string: unique identifier of the resource. cannot be null; length must be min=3 and max=32; allowed characters can be capital letters (A-Z), digits (0-9), hyphen (-); cannot start and/or end with an hyphen
caption	string: a title or brief explanation of the resource. cannot be null; length must be min=3 and max=128; all characters except line breaks are allowed.
description	string: full or long explanation of the resource. can be null or empty;
colorMap	ColorMap: defines the color values for the pixels of a raster image, as either color gradients, or a mapping of specific values to fixed colors.

ColorMap JSON representation

```
{
  “type”: string,
  “items”: [
     ColorMapItem
  ]
}
```

FIELD	DESCRIPTION
type	string: specifies the kind of ColorMap to use. cannot be null; there are three different types of ColorMap that can be specified: • INTERPOLATED is the default ColorMap type. It specifies that colors should be interpolated for values between the color map entries. • INTERVALS specifies that each interval defined by two entries is rendered using the color of the first (lowest-value) entry. No color interpolation is applied across the intervals. • VALUES specifies that only pixels with the selected entry quantity values are rendered. Pixels with other values are not rendered.
items	array: list of ColorMapItem objects that defines the ColorMap entries cannot be null nor empty;

ColorMapItem JSON representation

```
{
  “color”: string,
  “quantity”: double,
  “opacity”: double,
  “label”: string
}
```

FIELD	DESCRIPTION
color	string: specifies the color that must be applied to the pixel. cannot be null; value is denoted in standard hexadecimal RGB format (#RRGGBB).
quantity	string: specifies the value of raster pixel; cannot be null;
opacity	string: specifies the transparency that must be applied to the pixel. can be null; min value=0.0; max value=1.0; default=1.0
label	string: is used to provide text for graphical legends; can be null or empty

7.1. Create

To create a style, use the following POST request and include the authorization described in Authorizing Requests.

After a style has been created, on the DATA bucket will be stored the following files:

- <WORKSPACE_CODE>_<CODE>.json
  containing the JSON representation of the style

- <WORKSPACE_CODE>_<CODE>.sld
  containing the SLD representation of the style.

- <WORKSPACE_CODE>_<CODE>.png
  representing the style legend in image/png format

HTTP Request:

```
POST /catalog/workspaces/{w_code}/styles
```

Request Headers:

```
       Accept: application/json
 Content-Type: multipart/form-data
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
`w_code`	the unique code of the workspace in which you want to create the style

Request Body:

as the `Content-Type` of the request must be set to `multipart/form-data` the submitting form must be composed by two parts:

PART	DESCRIPTION
`style`	`object`: a JSON object containing the following `style` fields: • `workspaceCode` • `code` • `caption` • `description` The style object can be passed as a JSON string or as a JSON file; the `Content-Type` attribute of this part must be set equal to `application/json`.
`file`	`file`: a file containing the style's `ColorRamp` definition. two kind of file are allowed: • `SLD` (Styled Layer Descriptor) • JSON The `Content-Type` attribute of this part must be set equal to `application/octet-stream`

Response Body:

If successful (HTTP response code = **201**), the response body contains the newly created `style` resource.

Example:

create the STYLE.json with the following content:

```
{
  "workspaceCode": "ACME",
  "code": "LCC",
  "caption": "Land Cover Classification"
}
```

create the STYLE.sld file with the following content:

```
<?xml version="1.0" ?>
<sld:StyledLayerDescriptor version="1.0.0" xmlns="http://www.opengis.net/sld" xmlns:gml="http://www.opengis.net/gml"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:sld="http://www.opengis.net/sld">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>L3_AWA_LCC_12s1_irri_xLegend</sld:Name>
      <sld:Description>Generated by SLD4raster - https://cbsuygulama.wordpress.com/sld4raster</sld:Description>
      <sld:Title/>
      <sld:FeatureTypeStyle>
        <sld:Name/>
        <sld:Rule>
          <sld:RasterSymbolizer>
            <sld:Geometry>
              <ogc:PropertyName>grid</ogc:PropertyName>
            </sld:Geometry>
            <sld:Opacity>1</sld:Opacity>
            <sld:ColorMap type="values">
              <sld:ColorMapEntry color="#107020" label="1-Tree cover (dense)" opacity="1.0" quantity="1"/>
              <sld:ColorMapEntry color="#c2d581" label="4-Grassland" opacity="1.0" quantity="4"/>
              <sld:ColorMapEntry color="#aa1f28" label="7-Artificial" opacity="1.0" quantity="7"/>
              <sld:ColorMapEntry color="#00fad1" label="8-Wheat" opacity="1.0" quantity="8"/>
              <sld:ColorMapEntry color="#ff88ff" label="9-Maize" opacity="1.0" quantity="9"/>
              <sld:ColorMapEntry color="#f5c7c2" label="11-Vegetables" opacity="1.0" quantity="11"/>
              <sld:ColorMapEntry color="#eb9b31" label="12-Fallow" opacity="1.0" quantity="12"/>
              <sld:ColorMapEntry color="#73fc85" label="13-Orchard " opacity="1.0" quantity="13"/>
              <sld:ColorMapEntry color="#647f9e" label="17-Wetland" opacity="1.0" quantity="17"/>
              <sld:ColorMapEntry color="#b57060" label="18-Shrubland" opacity="1.0" quantity="18"/>
              <sld:ColorMapEntry color="#294ead" label="19-Water" opacity="1.0" quantity="19"/>
              <sld:ColorMapEntry color="#c2b6cc" label="21-Other crop" opacity="1.0" quantity="21"/>
              <sld:ColorMapEntry color="#e8e5d0" label="30-Bare, not vegetated" opacity="1.0" quantity="30"/>
              <sld:ColorMapEntry color="#00fad1" label="108-Irrigated wheat" opacity="1.0" quantity="108"/>
              <sld:ColorMapEntry color="#ff00ff" label="109-Irrigated maize" opacity="1.0" quantity="109"/>
              <sld:ColorMapEntry color="#f59f94" label="111-Irrigated vegetables" opacity="1.0" quantity="111"/>
              <sld:ColorMapEntry color="#0cf72c" label="113-Irrigated orchard" opacity="1.0" quantity="113"/>
              <sld:ColorMapEntry color="#a99eb2" label="121-Irrigated other crops" opacity="1.0" quantity="121"/>
              <sld:ColorMapEntry color="#a125d6" label="122-Irrigated sugar cane" opacity="1.0" quantity="122"/>
            </sld:ColorMap>
          </sld:RasterSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>
```

execute the request.

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...
```

```
# create a new style
curl -X POST -H "Content-Type: multipart/form-data" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -F "style=@STYLE.json; type=application/json" -F
"file=@STYLE.sld" "${BASE_URL}/catalog/workspaces/ACME/styles"

# RESPONSE:
{
  "requestId": "a187c7ab-5822-43ad-a837-8c7ea6410553",
  "timestamp": 1669742218644,
  "runtime": 414,
  "status": 201,
  "message": "Created",
  "response": {
    "workspaceCode": "ACME",
    "code": "LCC",
    "caption": "Land Cover Classification",
    "description": null,
    "colorMap": {
      "type": "VALUES",
      "items": [
        {
          "color": "#107020",
          "quantity": 1.0,
          "opacity": 1.0,
          "label": "1-Tree cover (dense)"
        },
        {
          "color": "#c2d581",
          "quantity": 4.0,
          "opacity": 1.0,
          "label": "4-Grassland"
        },
        {
          "color": "#aa1f28",
          "quantity": 7.0,
          "opacity": 1.0,
          "label": "7-Artificial"
        },
        {
          "color": "#00fad1",
          "quantity": 8.0,
          "opacity": 1.0,
          "label": "8-Wheat"
        },
        {
          "color": "#ff88ff",
          "quantity": 9.0,
          "opacity": 1.0,
          "label": "9-Maize"
        },
        {
          "color": "#f5c7c2",
          "quantity": 11.0,
          "opacity": 1.0,
          "label": "11-Vegetables"
        },
        {
          "color": "#eb9b31",
          "quantity": 12.0,
          "opacity": 1.0,
          "label": "12-Fallow"
```

```
        },
        {
          "color": "#73fc85",
          "quantity": 13.0,
          "opacity": 1.0,
          "label": "13-Orchard"
        },
        {
          "color": "#647f9e",
          "quantity": 17.0,
          "opacity": 1.0,
          "label": "17-Wetland"
        },
        {
          "color": "#b57060",
          "quantity": 18.0,
          "opacity": 1.0,
          "label": "18-Shrubland"
        },
        {
          "color": "#294ead",
          "quantity": 19.0,
          "opacity": 1.0,
          "label": "19-Water"
        },
        {
          "color": "#c2b6cc",
          "quantity": 21.0,
          "opacity": 1.0,
          "label": "21-Other crop"
        },
        {
          "color": "#e8e5d0",
          "quantity": 30.0,
          "opacity": 1.0,
          "label": "30-Bare, not vegetated"
        },
        {
          "color": "#00fad1",
          "quantity": 108.0,
          "opacity": 1.0,
          "label": "108-Irrigated wheat"
        },
        {
          "color": "#ff00ff",
          "quantity": 109.0,
          "opacity": 1.0,
          "label": "109-Irrigated maize"
        },
        {
          "color": "#f59f94",
          "quantity": 111.0,
          "opacity": 1.0,
          "label": "111-Irrigated vegetables"
        },
        {
          "color": "#0cf72c",
          "quantity": 113.0,
          "opacity": 1.0,
          "label": "113-Irrigated orchard"
        },
        {
```

```
          "color": "#a99eb2",
          "quantity": 121.0,
          "opacity": 1.0,
          "label": "121-Irrigated other crops"
        },
        {
          "color": "#a125d6",
          "quantity": 122.0,
          "opacity": 1.0,
          "label": "122-Irrigated sugar cane"
        }
      ]
    },
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC"
      },
      {
        "rel": "sld",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC/sld"
      },
      {
        "rel": "legend",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC/legend"
      }
    ]
  }
}
```


7.2. List

Retrieves a list of `style` resources (paginated).

Authentication/Authorization is not required

HTTP Request:

```
GET /catalog/workspaces/{w_code}/styles
```

Request Headers:

```
Accept: application/json
```

Path Parameters:

PARAMETER	DESCRIPTION
`w_code`	the unique code of the workspace to which the styles list (page) belongs

Query Parameters

PARAMETER	DESCRIPTION
`view`	**`optional`** the level of the details of the resources returned by the response: • `overview` - only the main fields (default). • `details` - all the fields except the hidden ones • `hidden` - all the fields
`size`	**`optional`** max number of resource items per page. min value=1; max value=100; default=20
`page`	**`optional`** number of the page; default=0 → first page

`sort`	**`optional`**
	sort the resource items, in ascending or descending order, according to the field(s) specified. Note that page/size (paginated request) parameters are applied after the full result-set has been sorted.
	
	`syntax`:
	`sort=field[:{ASC|DESC}][,field[:{ASC|DESC}]][,...]`
	
	• `field` must not be complex type (arrays or objects)
	• ASC and DESC are case insensitive (default=ASC)
	• no space characters are allowed
	
	Example:
	`sort=caption,code:desc`

Response Body:

If successful (HTTP response code = **`200`**), the response body contains the list of `style` resource items.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# list the workspace's styles (paginated)
curl -X GET -H "Accept: application/json"
"${BASE_URL}/catalog/workspaces/ACME/styles"

# RESPONSE:
{
  "requestId": "6383f83a-d8a3-45b9-8d0a-3436bff2c616",
  "timestamp": 1669745599398,
  "runtime": 8,
  "status": 200,
  "message": "OK",
  "response": {
    "size": 20,
    "number": 0,
    "hasPrevious": false,
    "hasNext": false,
    "items": [
      {
        "workspaceCode": "ACME",
        "code": "LCC",
        "caption": "Land Cover Classification",
        "links": [
          {
            "rel": "self",
            "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC"
          }
        ]
      },
      . . .
    ],
    "links": [
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles"
      }
    ]
  }
}
```

7.3. Get (JSON)

Retrieves an instance of `style` resource in JSON format.

Authentication/Authorization is not required

HTTP Request:

```
GET /catalog/workspaces/{w_code}/styles/{s_code}
```

Request Headers:

```
Accept: application/json
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the style belongs
s_code	the unique code of the style you want to retrieve.

Query Parameters:

PARAMETER	DESCRIPTION
view	**optional** the level of the details of the resources returned by the response: • overview - only the main fields. • details - all the fields except the hidden ones (default) • hidden - all the fields

Response Body:

If successful (HTTP response code = **200**), the response body contains the selected instance of style resource.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# get a selected style resource in JSON format
curl -X GET -H "Accept: application/json"
"${BASE_URL}/catalog/workspaces/ACME/styles/LCC?view=overview"

# RESPONSE:
{
  "requestId": "b22905c9-78d7-4bf5-9925-62bf91f1bf87",
  "timestamp": 1669746790676,
  "runtime": 7,
  "status": 200,
  "message": "OK",
```

```
  "response": {
    "workspaceCode": "ACME",
    "code": "LCC",
    "caption": "Land Cover Classification",
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC?view=over
view"
      },
      {
        "rel": "sld",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC/sld"
      },
      {
        "rel": "legend",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC/legend"
      }
    ]
  }
}
```


7.4. Get (SLD)

Retrieves an instance of `style` resource in SLD format.

Authentication/Authorization is not required

HTTP Request:

```
GET /catalog/workspaces/{w_code}/styles/{s_code}/sld
```

Request Headers:

```
Accept: application/xml
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the style belongs
s_code	the unique code of the style you want to retrieve.

Response Body:

If successful (HTTP response code = **200**), the response body is the selected instance of style resource translated into SLD format.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# get a selected style resource in JSON format
curl -X GET -H "Accept: application/xml"
"${BASE_URL}/catalog/workspaces/ACME/styles/LCC/sld"

# RESPONSE:
<?xml version="1.0" encoding="UTF-8"?>
<StyledLayerDescriptor xmlns="http://www.opengis.net/sld" xmlns:gml="http://www.opengis.net/gml"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:sld="http://www.opengis.net/sld" version="1.0.0">
  <UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>ACME_LCC</sld:Name>
      <sld:Title>Land Cover Classification</sld:Title>
      <sld:FeatureTypeStyle>
        <sld:Rule>
          <sld:RasterSymbolizer>
            <sld:ColorMap type="values">
              <sld:ColorMapEntry color="#107020" opacity="1.0" quantity="1.0" label="1-Tree cover (dense)"/>
              <sld:ColorMapEntry color="#c2d581" opacity="1.0" quantity="4.0" label="4-Grassland"/>
              <sld:ColorMapEntry color="#aa1f28" opacity="1.0" quantity="7.0" label="7-Artificial"/>
              <sld:ColorMapEntry color="#00fad1" opacity="1.0" quantity="8.0" label="8-Wheat"/>
              <sld:ColorMapEntry color="#ff88ff" opacity="1.0" quantity="9.0" label="9-Maize"/>
              <sld:ColorMapEntry color="#f5c7c2" opacity="1.0" quantity="11.0" label="11-Vegetables"/>
              <sld:ColorMapEntry color="#eb9b31" opacity="1.0" quantity="12.0" label="12-Fallow"/>
              <sld:ColorMapEntry color="#73fc85" opacity="1.0" quantity="13.0" label="13-Orchard"/>
              <sld:ColorMapEntry color="#647f9e" opacity="1.0" quantity="17.0" label="17-Wetland"/>
              <sld:ColorMapEntry color="#b57060" opacity="1.0" quantity="18.0" label="18-Shrubland"/>
              <sld:ColorMapEntry color="#294ead" opacity="1.0" quantity="19.0" label="19-Water"/>
              <sld:ColorMapEntry color="#c2b6cc" opacity="1.0" quantity="21.0" label="21-Other crop"/>
              <sld:ColorMapEntry color="#e8e5d0" opacity="1.0" quantity="30.0" label="30-Bare, not vegetated"/>
              <sld:ColorMapEntry color="#00fad1" opacity="1.0" quantity="108.0" label="108-Irrigated wheat"/>
              <sld:ColorMapEntry color="#ff00ff" opacity="1.0" quantity="109.0" label="109-Irrigated maize"/>
              <sld:ColorMapEntry color="#f59f94" opacity="1.0" quantity="111.0" label="111-Irrigated vegetables"/>
              <sld:ColorMapEntry color="#0cf72c" opacity="1.0" quantity="113.0" label="113-Irrigated orchard"/>
              <sld:ColorMapEntry color="#a99eb2" opacity="1.0" quantity="121.0" label="121-Irrigated other crops"/>
              <sld:ColorMapEntry color="#a125d6" opacity="1.0" quantity="122.0" label="122-Irrigated sugar cane"/>
            </sld:ColorMap>
          </sld:RasterSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </UserLayer>
</StyledLayerDescriptor>
```

7.5. Get (PNG Legend)

Retrieves the graphic legend (image) related to the `style` resource in PNG format.

Authentication/Authorization is not required

HTTP Request:

```
GET /catalog/workspaces/{w_code}/styles/{s_code}/legend
```

Request Headers:

```
Accept: image/png
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the style belongs
s_code	the unique code of the style of which you want to retrieve the legend.

Query Parameters:

PARAMETER	DESCRIPTION
margin	integer: optional; default=20. the thickness of the transparent area around the content expressed in pixels.
horizontalSpace	integer: optional; default=10. the space between the color-box and the label (if defined); expressed in pixels.
verticalSpace	integer: optional; the space between two color-map entries expressed in pixels; for INTERPOLATED color-maps the value is always set equal to 0, otherwise the default value is equal to 8
colorBoxWith	integer: optional; the width of the color box expressed in pixels; for INTERPOLATED color-maps the default value is set equal to 33, otherwise is equal to 15.

colorBoxHeight	integer: optional; default=25. the height of the color box expressed in pixels.
fontName	string: optional; default=Dialog. the name of the font applied to draw style labels.
fontSize	integer: optional; default=14 the size of the font to draw style labels
fontColor	string: optional; default=#ffffff → white the color of the font to draw style labels expressed in standard hexadecimal RGB format (#RRGGBB).
backGroundColor	string: optional; default=#333333. the back-groud color of the legend expressed in standard hexadecimal RGB format (#RRGGBB).

Response Body:

If successful (HTTP response code = **200**), the response body is a PNG image representing the style's legend.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# get the legend (saved as STYLE.png file)
curl -X GET -H "Accept: image/png" -o "STYLE.png"
"${BASE_URL}/catalog/workspaces/ACME/styles/LCC/legend"

# RESPONSE
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 16722  100 16722    0     0  56541      0 --:--:-- --:--:-- --:--:-- 57267
```

STYLE . png file content:

7.6. Update

To update a style, use the following PUT request and include the authorization described in Authorizing Requests.

HTTP Request:

```
PUT /catalog/workspaces/{w_code}/styles/{s_code}
```

Request Headers:

```
       Accept: application/json
 Content-Type: multipart/form-data
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
`w_code`	the unique code of the workspace to which the style belongs
`s_code`	the unique code of the style you want to to update

Request Body:

as the `Content-Type` of the request must be set to `multipart/form-data` the submitting form must be composed by two parts:

PART	DESCRIPTION
`style`	`object`: a JSON object containing the following `style` fields: • `workspaceCode` • `code` • `caption` • `description` The style object can be passed as a JSON string or as a JSON file; the `Content-Type` attribute of this part must be set equal to `application/json`.
`file`	`file`: a file containing the style's `ColorRamp` definition. two kind of file are allowed: • SLD (Styled Layer Descriptor) • JSON The `Content-Type` attribute of this part must be set equal to `application/octet-stream`

Response Body:

If successful (HTTP response code = **200**), the response body contains the updated `style` resource.

Example:

update the STYLE.json file (description field added):

```
{
  "workspaceCode": "ACME",
  "code": "LCC",
  "caption": "Land Cover Classification",
  “description”: “a detailed classification with information on the crops representing at least 10% of the area.”
}
```

update the STYLE.sld file (first and last color have been updated):

```
<?xml version="1.0" ?>
<sld:StyledLayerDescriptor version="1.0.0" xmlns="http://www.opengis.net/sld" xmlns:gml="http://www.opengis.net/gml"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:sld="http://www.opengis.net/sld">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>L3_AWA_LCC_12s1_irri_xLegend</sld:Name>
      <sld:Description>Generated by SLD4raster - https://cbsuygulama.wordpress.com/sld4raster</sld:Description>
      <sld:Title/>
      <sld:FeatureTypeStyle>
        <sld:Name/>
        <sld:Rule>
          <sld:RasterSymbolizer>
            <sld:Geometry>
              <ogc:PropertyName>grid</ogc:PropertyName>
            </sld:Geometry>
            <sld:Opacity>1</sld:Opacity>
            <sld:ColorMap type="values">
              <sld:ColorMapEntry color="#107020" label="1-Tree cover (dense)" opacity="1.0" quantity="1"/>
              <sld:ColorMapEntry color="#c2d581" label="4-Grassland" opacity="1.0" quantity="4"/>
              <sld:ColorMapEntry color="#aa1f28" label="7-Artificial" opacity="1.0" quantity="7"/>
              <sld:ColorMapEntry color="#00fad1" label="8-Wheat" opacity="1.0" quantity="8"/>
              <sld:ColorMapEntry color="#ff88ff" label="9-Maize" opacity="1.0" quantity="9"/>
              <sld:ColorMapEntry color="#f5c7c2" label="11-Vegetables" opacity="1.0" quantity="11"/>
              <sld:ColorMapEntry color="#eb9b31" label="12-Fallow" opacity="1.0" quantity="12"/>
              <sld:ColorMapEntry color="#73fc85" label="13-Orchard " opacity="1.0" quantity="13"/>
              <sld:ColorMapEntry color="#647f9e" label="17-Wetland" opacity="1.0" quantity="17"/>
              <sld:ColorMapEntry color="#b57060" label="18-Shrubland" opacity="1.0" quantity="18"/>
              <sld:ColorMapEntry color="#294ead" label="19-Water" opacity="1.0" quantity="19"/>
              <sld:ColorMapEntry color="#c2b6cc" label="21-Other crop" opacity="1.0" quantity="21"/>
              <sld:ColorMapEntry color="#e8e5d0" label="30-Bare, not vegetated" opacity="1.0" quantity="30"/>
              <sld:ColorMapEntry color="#00fad1" label="108-Irrigated wheat" opacity="1.0" quantity="108"/>
              <sld:ColorMapEntry color="#ff00ff" label="109-Irrigated maize" opacity="1.0" quantity="109"/>
              <sld:ColorMapEntry color="#f59f94" label="111-Irrigated vegetables" opacity="1.0" quantity="111"/>
              <sld:ColorMapEntry color="#0cf72c" label="113-Irrigated orchard" opacity="1.0" quantity="113"/>
              <sld:ColorMapEntry color="#a99eb2" label="121-Irrigated other crops" opacity="1.0" quantity="121"/>
              <sld:ColorMapEntry color="#a125d6" label="122-Irrigated sugar cane" opacity="1.0" quantity="122"/>
            </sld:ColorMap>
          </sld:RasterSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>
```

execute the request.

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# update an existing style
curl -X PUT -H "Content-Type: multipart/form-data" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -F "style=@STYLE.json; type=application/json" -F
"file=@STYLE.sld" "${BASE_URL}/catalog/workspaces/ACME/styles/LCC"

# RESPONSE:
{
  "requestId": "cfb49ae1-f635-43f1-925d-c58feb821cc6",
  "timestamp": 1669805554814,
  "runtime": 435,
  "status": 200,
  "message": "OK",
  "response": {
    "workspaceCode": "ACME",
    "code": "LCC",
    "caption": "Land Cover Classification",
    "description": "a detailed classification with information on the the crops
representing at least 10% of the area.",
    "colorMap": {
      "type": "VALUES",
      "items": [
        {
          "color": "#006600",
          "quantity": 1.0,
          "opacity": 1.0,
          "label": "1-Tree cover (dense)"
        },
        {
          "color": "#c2d581",
          "quantity": 4.0,
          "opacity": 1.0,
          "label": "4-Grassland"
        },
        {
          "color": "#aa1f28",
          "quantity": 7.0,
          "opacity": 1.0,
          "label": "7-Artificial"
        },
        {
          "color": "#00fad1",
          "quantity": 8.0,
          "opacity": 1.0,
          "label": "8-Wheat"
        },
        {
          "color": "#ff88ff",
          "quantity": 9.0,
          "opacity": 1.0,
          "label": "9-Maize"
        },
        {
          "color": "#f5c7c2",
          "quantity": 11.0,
          "opacity": 1.0,
```

```
          "label": "11-Vegetables"
        },
        {
          "color": "#eb9b31",
          "quantity": 12.0,
          "opacity": 1.0,
          "label": "12-Fallow"
        },
        {
          "color": "#73fc85",
          "quantity": 13.0,
          "opacity": 1.0,
          "label": "13-Orchard"
        },
        {
          "color": "#647f9e",
          "quantity": 17.0,
          "opacity": 1.0,
          "label": "17-Wetland"
        },
        {
          "color": "#b57060",
          "quantity": 18.0,
          "opacity": 1.0,
          "label": "18-Shrubland"
        },
        {
          "color": "#294ead",
          "quantity": 19.0,
          "opacity": 1.0,
          "label": "19-Water"
        },
        {
          "color": "#c2b6cc",
          "quantity": 21.0,
          "opacity": 1.0,
          "label": "21-Other crop"
        },
        {
          "color": "#e8e5d0",
          "quantity": 30.0,
          "opacity": 1.0,
          "label": "30-Bare, not vegetated"
        },
        {
          "color": "#00fad1",
          "quantity": 108.0,
          "opacity": 1.0,
          "label": "108-Irrigated wheat"
        },
        {
          "color": "#ff00ff",
          "quantity": 109.0,
          "opacity": 1.0,
          "label": "109-Irrigated maize"
        },
        {
          "color": "#f59f94",
          "quantity": 111.0,
          "opacity": 1.0,
          "label": "111-Irrigated vegetables"
        },
```

```
        {
          "color": "#0cf72c",
          "quantity": 113.0,
          "opacity": 1.0,
          "label": "113-Irrigated orchard"
        },
        {
          "color": "#a99eb2",
          "quantity": 121.0,
          "opacity": 1.0,
          "label": "121-Irrigated other crops"
        },
        {
          "color": "#9900cc",
          "quantity": 122.0,
          "opacity": 1.0,
          "label": "122-Irrigated sugar cane"
        }
      ]
    },
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC"
      },
      {
        "rel": "sld",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC/sld"
      },
      {
        "rel": "legend",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC/legend"
      }
    ]
  }
}
```


7.7. Delete

To delete a style, use the following DELETE request and include the authorization described in Authorizing Requests.

HTTP Request:

```
DELETE /catalog/workspaces/{w_code}/styles/{s_code}
```

Request Headers:

```
      Accept: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the style belongs
s_code	the unique code of the style you want to to delete

Response Body:

If successful (HTTP response code = **200**), the response body contains the deleted style resource.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# delete an existing style
curl -X DELETE -H "Accept: application/json" -H "Authorization: Bearer ${ID_TOKEN}"
"${BASE_URL}/catalog/workspaces/ACME/styles/LCC"

# RESPONSE:
{
  "requestId": "022f4b4d-72c9-4cc1-896e-a75ad72b4fbc",
  "timestamp": 1669806049046,
  "runtime": 326,
  "status": 200,
  "message": "OK",
  "response": {
    "workspaceCode": "ACME",
    "code": "LCC",
    "caption": "Land Cover Classification",
    "description": "a detailed classification with information on the the crops
representing at least 10% of the area.",
    "colorMap": {
      "type": "VALUES",
      "items": [
        {
          "color": "#006600",
          "quantity": 1.0,
          "opacity": 1.0,
          "label": "1-Tree cover (dense)"
        },
        {
```

```
          "color": "#c2d581",
          "quantity": 4.0,
          "opacity": 1.0,
          "label": "4-Grassland"
        },
        {
          "color": "#aa1f28",
          "quantity": 7.0,
          "opacity": 1.0,
          "label": "7-Artificial"
        },
        {
          "color": "#00fad1",
          "quantity": 8.0,
          "opacity": 1.0,
          "label": "8-Wheat"
        },
        {
          "color": "#ff88ff",
          "quantity": 9.0,
          "opacity": 1.0,
          "label": "9-Maize"
        },
        {
          "color": "#f5c7c2",
          "quantity": 11.0,
          "opacity": 1.0,
          "label": "11-Vegetables"
        },
        {
          "color": "#eb9b31",
          "quantity": 12.0,
          "opacity": 1.0,
          "label": "12-Fallow"
        },
        {
          "color": "#73fc85",
          "quantity": 13.0,
          "opacity": 1.0,
          "label": "13-Orchard"
        },
        {
          "color": "#647f9e",
          "quantity": 17.0,
          "opacity": 1.0,
          "label": "17-Wetland"
        },
        {
          "color": "#b57060",
          "quantity": 18.0,
          "opacity": 1.0,
          "label": "18-Shrubland"
        },
        {
          "color": "#294ead",
          "quantity": 19.0,
          "opacity": 1.0,
          "label": "19-Water"
        },
        {
          "color": "#c2b6cc",
          "quantity": 21.0,
```

```
          "opacity": 1.0,
          "label": "21-Other crop"
        },
        {
          "color": "#e8e5d0",
          "quantity": 30.0,
          "opacity": 1.0,
          "label": "30-Bare, not vegetated"
        },
        {
          "color": "#00fad1",
          "quantity": 108.0,
          "opacity": 1.0,
          "label": "108-Irrigated wheat"
        },
        {
          "color": "#ff00ff",
          "quantity": 109.0,
          "opacity": 1.0,
          "label": "109-Irrigated maize"
        },
        {
          "color": "#f59f94",
          "quantity": 111.0,
          "opacity": 1.0,
          "label": "111-Irrigated vegetables"
        },
        {
          "color": "#0cf72c",
          "quantity": 113.0,
          "opacity": 1.0,
          "label": "113-Irrigated orchard"
        },
        {
          "color": "#a99eb2",
          "quantity": 121.0,
          "opacity": 1.0,
          "label": "121-Irrigated other crops"
        },
        {
          "color": "#9900cc",
          "quantity": 122.0,
          "opacity": 1.0,
          "label": "122-Irrigated sugar cane"
        }
      ]
    }
  }
}
```

8. Map

In GISMGR a map is formed by metadata that describes the resource and a *single raster layer*, namely a layer having, as its data source, a spatial data model that defines space as an array of equally sized cells arranged in rows and columns, generally stored on disk or in a (GIS) database.

A map describes objects, events or other features with a location on or near the surface of the earth combining location information (usually coordinates on the earth) and attribute information (the characteristics of the object, event or phenomena concerned) with temporal information (the time or life span at which the location and attributes exist).

As geospatial data is most useful when it can be discovered, shared, analyzed and used in combination with traditional business data, GISMGR handles different input formats, continuous or categorized rasters, translates to COG format and finally stores/shares them (and their related metadata as JSON files) on workspaces' DATA buckets.

Map JSON representation

```
{
  "workspaceCode": string,
  "code": string,
  "caption": string,
  "description": string,
  "styleCode": string,
  "extensions": [
    string
  ],
  "measureCaption": string,
  "measureUnit": string,
  "scale": double,
  "offset": double,
  "flags": object,
  "classes": object,
  "translate": [
    TranslateItem
  ],
  "additionalInfo": object,
  "tags": [
     string
  ]
}
```

FIELD	DESCRIPTION
workspaceCode	string: unique code of the workspace to which the resource belongs. cannot be null; length must be min=3 and max=32; allowed characters can be capital letters (A-Z), digits (0-9), hyphen (-); cannot start and/or end with an hyphen
code	string: unique identifier of the resource. cannot be null; length must be min=3 and max=32; allowed characters can be capital letters (A-Z), digits (0-9), hyphen (-); cannot start and/or end with an hyphen
caption	string: a title or brief explanation of the resource. cannot be null; length must be min=3 and max=128; all characters except line breaks are allowed.
description	string: full or long explanation of the resource. can be null or empty;
styleCode	string: unique code of a previously defined style that must be applied for rendering the map's raster file. can be null; length must be min=3 and max=32; allowed characters can be capital letters (A-Z), digits (0-9), hyphen (-); cannot start and/or end with an hyphen
extensions	array: list of strings representing the expected raster dataset extensions. raster data can be formed by one or more files; set the expected extensions of the files that represent the full raster data, putting as first extension the main one. cannot be null (expected one extension at least); if formed by two or more files the file names must differ only in the extension part.
measureCaption	string: a title or brief explanation of the phenomenon being observed. can be null or empty
measureUnit	string: unit symbols of measurement. unit symbols are typically shortened versions of the unit name. can be null or empty; if the raster represents classes or categories unit value must be set equal to "class"
scale	double: slope of the slope-intercept form of linear equation to be applied to pixel value to to get the actual value. cannot be null; default 1;
offset	double: intercept of the slope-intercept form of linear equation to be applied to pixel value to to get the actual value. cannot be null; default 0;
flags	object: a JSON object representing pixel values with specific meaning.

	can be null; if provided for each pixel value the `caption` is mandatory, while `description` is optional. **example:** `{ "250": { "caption": "off season" }, "251": { "caption": "no seasonality", } "253": { "caption": "insufficient data", } "254": { "caption": "no cropland / no grassland", } }`
`classes`	`object`: a JSON object representing pixel values and their meaning. must be provided only if the measureUnit=`class`; for each pixel value the `caption` is mandatory, while `description` is optional. **example:** `{ "1": { "caption": "Tree cover (dense) ", "description": "Tree cover with a canopy cover >70% and height exceeding 5m." }, "4": { "caption": "Grassland", "description": "Mainly grass cover with woody vegetation, canopy cover <15%" }, . . . "255": { "caption": "no data", "description": "no data" } }`
`translate`	`array`: list of `TranslateItem` that define the *input raster to COG* format conversion process. can be null or empty; if not provided the default one is applied: `[ { "format": "GTiff", "options": [ "INTERLEAVE=BAND", "TILED=YES", "COMPRESS=LZW" ] } ]`

additionalInfo	object: additional information about the resource. can be null; a freely structured JSON object provided by the data owner to better describe the resource
tags	array: list of string objects; can be null or empty; a **tag** is a keyword or term assigned to a piece of information to describe it.

TranslateItem JSON representation

```
{
  "subDataset": string,
  "band": string,
  "format": string,
  "dataType": string,
  "noDataValue": string,
  "scale": [
     string
  ],
  "srs": string,
  "extent": [
     string
  ],
  "gridSubWindow": [
     string
  ]
  "srsSubWindow": [
     string
  ]
  "options": [
     string
  ]
}
```

FIELD	DESCRIPTION
subDataset	string: selects the name of the input sub-dataset. (some raster drivers, like NetCDF or HDF5, support multidimensional datasets)
band	string: selects an input band for output; bands are numbered from 1.

format	string: selects the output format (raster driver); use the short format/driver name; for the full list of supported raster drivers see here
dataType	string: forces the output raster bands to have a specific data type supported by the driver, which may be one of the following: • Byte • Int8 • UInt16 • Int16 • UInt32 • Int32 • UInt64 • Int64 • Float32 • Float64 • CInt16 • CInt32 • CFloat32 • CFloat64
noDataValue	string: assigns a specified no-data value to the output file. if you want to remove or avoid to assign a no-data value you can set this option equal to none. Note that this option will not update any pixel value.
scale	array: array of 4 elements (string): • src_min - input raster min value • src_max - input raster max value • dst_min - output raster min value • dst_max - output raster max value rescales the input pixel values from the range {src_min..src_max} to the {dst_min..dst_max} one
srs	string: assigns a specified projection to the output file the spatial reference systems (projection) can be expressed using an EPSG code or a PROJ.4 declaration string. **examples**: EPSG:4326 or +proj=longlat +datum=WGS84 +no_defs +type=crs
extent	array: array of 4 elements (string):

	• `ulx` - upper left corner x coordinate • `uly` - upper left corner y coordinate • `lrx` - lower right corner x coordinate • `lry` - lower right corner y coordinate sets/assigns the georeferenced bounds of the output file. Note this option does not cause reprojection.
`gridSubWindow`	`array`: array of 4 elements (`string`): • `xoff` - grid window's upper left corner x coordinate (0-indexed) • `yoff` - grid window's upper left corner y coordinate (0-indexed) • `xsize` - window's width size expressed in pixels • `ysize` - window's height size expressed in pixels selects a subwindow from the source image for copying based on pixel/line (column/row) location.
`srsSubWindow`	`array`: array of 4 elements (`string`): • `ulx` - window's upper left corner x coordinate • `uly` - window's upper left corner y coordinate • `lrx` - window's lower right corner x coordinate • `lry` - window's lower right corner y coordinate selects a subwindow from the source image for copying based on georeferenced coordinates expressed in the SRS of the input raster Note that the SRS window will be previously translated to grid coordinates.
`options`	`array`: array of elements expressed in the `NAME=VALUE` form, where: • NAME is the name of the creation option • VALUE is the selected value to be assigned to the option defines the creation options for the output file. Many formats/drivers have one or more creation options that can be used to control particulars about the output file created. For instance, the GeoTIFF driver supports creation options to control compression, and whether the file should be tiled. The creation options available vary by format driver, and some simple formats have no creation options at all. See Raster drivers format specific documentation for legal creation options for each format. For the full list of supported raster drivers see here

8.1. Create

To create a map, use the following POST request and include the authorization described in Authorizing Requests.

While the map creation process, a new folder will be created on the workspace's UPLOAD bucket:

```
/MAP/<CODE>
```

where `CODE` is the unique identifier assigned to the map.

Then, to bind a raster layer to the map, the files (one or more) that form the raster dataset must be uploaded to the above folder.

After a raster layer has been bound to the map, in the workspace's DATA bucket under the `/DATA/<WORKSPACE_CODE>/MAP` folder, the following files will be stored:

- `<WORKSPACE_CODE>_<CODE>.tif`
  the normalized raster layer in COG format (as result of the ingestion process)

- `<WORKSPACE_CODE>_<CODE>.json`
  containing the JSON representation of the map and of the related bound raster

- `<WORKSPACE_CODE>_<CODE>.html`
  a working preview of the map contained in a standalone HTML file

HTTP Request:

```
POST /catalog/workspaces/{w_code}/maps
```

Request Headers:

```
       Accept: application/json
 Content-Type: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace in which you want to create the map

Request Body:

The request body is an instance of `map` resource.

Response Body:

If successful (HTTP response code = **201**), the response body contains the newly created instance of `map` resource.

Example:

prepare a MAP.json file with the following content:

```
{
  "workspaceCode": "ACME",
  "code": "AWA-LCC",
  "caption": "Land Cover Classification (Awash, Ethiopia)",
  "description": "This land cover dataset at sub-national scale shows a detailed
classification with information on the the crops representing at least 10% of the
area.",
  "extensions": [
    ".tif"
  ],
  "styleCode": "LCC",
  "measureCaption": "Land Cover Classification",
  "measureUnit": "class",
  "classes": {
    "1": {
      "caption": "Tree cover (closed)"
    },
    "4": {
      "caption": "Grassland"
    },
    "11": {
      "caption": "Vegetables"
    },
    "12": {
      "caption": "Fallow"
    },
    "18": {
      "caption": "Shrubland"
    },
    "19": {
      "caption": "Water"
    },
    "21": {
      "caption": "Unknown mixed crops"
    },
    "30": {
      "caption": "Non vegetation (reclass)"
    },
    "111": {
      "caption": "Vegetables irrigated"
    },
    "113": {
      "caption": "Orchard irrigated (closed)"
    },
```

```
    "121": {
      "caption": "Unknown mixed crops irrigated"
    },
    "122": {
      "caption": "Sugarcane irrigated"
    },
    "22.00": {
      "caption": "Sugarcane"
    }
  },
  "tags": [
    "ETHIOPIA",
    "AWASH",
    "LAND"
  ]
}
```

execute the request:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# create
curl -X POST -H "Content-Type: application/json" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -d @MAP.json
"${BASE_URL}/catalog/workspaces/ACME/maps"

# RESPONSE
{
  "requestId": "9cf71a83-bff2-451d-9d73-f83c9c2e3b64",
  "timestamp": 1670411628777,
  "runtime": 195,
  "status": 201,
  "message": "Created",
  "response": {
    "workspaceCode": "ACME",
    "code": "AWA-LCC",
    "caption": "Land Cover Classification (Awash, Ethiopia)",
    "description": "This land cover dataset at sub-national scale ...",
    "styleCode": "LCC",
    "extensions": [
      ".tif"
    ],
    "measureCaption": "Land Cover Classification",
    "measureUnit": "class",
    "slope": 1.0,
    "intercept": 0.0,
    "flags": null,
    "classes": {
      "1": {
        "caption": "Tree cover (closed)"
      },
      "4": {
        "caption": "Grassland"
      },
      "11": {
        "caption": "Vegetables"
      },
```

```
      "12": {
        "caption": "Fallow"
      },
      "18": {
        "caption": "Shrubland"
      },
      "19": {
        "caption": "Water"
      },
      "21": {
        "caption": "Unknown mixed crops"
      },
      "30": {
        "caption": "Non vegetation (reclass)"
      },
      "111": {
        "caption": "Vegetables irrigated"
      },
      "113": {
        "caption": "Orchard irrigated (closed)"
      },
      "121": {
        "caption": "Unknown mixed crops irrigated"
      },
      "122": {
        "caption": "Sugarcane irrigated"
      },
      "22.00": {
        "caption": "Sugarcane"
      }
    },
    "translate": [
      {
        "subDataset": null,
        "band": null,
        "format": "GTiff",
        "dataType": null,
        "noDataValue": null,
        "scale": null,
        "srs": null,
        "extent": null,
        "gridSubWindow": null,
        "srsSubWindow": null,
        "options": [
          "COMPRESS=LZW",
          "INTERLEAVE=BAND",
          "TILED=YES"
        ]
      }
    ],
    "additionalInfo": null,
    "tags": [
      "AWASH",
      "ETHIOPIA",
      "LAND"
    ],
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps"
      },
```

```
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/AWA-LCC"
      },
      {
        "rel": "raster",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/AWA-LCC/raster"
      }
    ]
  }
}
```


8.2. List

Retrieves a list of `map` resources (paginated or sliced).

Authentication/Authorization is not required

HTTP Request:

`GET /catalog/workspaces/{w_code}/maps`

Request Headers:

`Accept: application/json`

Path Parameters:

PARAMETER	DESCRIPTION
`w_code`	the unique code of the workspace to which the maps list belongs

Query Parameters:

The default query parameters apply.

Response Body:

If successful (HTTP response code = **`200`**), the response body contains an array of `map` items.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# list of workspace's maps
curl -X GET -H "Accept: application/json" "${BASE_URL}/catalog/workspaces/ACME/maps"

# RESPONSE:
{
  "requestId": "1fa682e9-23ec-4e82-a215-27df0e8edb25",
  "timestamp": 1670412586404,
  "runtime": 11,
  "status": 200,
  "message": "OK",
  "response": {
    "size": 20,
    "number": 0,
    "hasPrevious": false,
    "hasNext": false,
    "items": [
      {
        "workspaceCode": "ACME",
        "code": "AWA-LCC",
        "caption": "Land Cover Classification (Awash, Ethiopia)",
        "description": "This land cover dataset at sub-national scale shows a
detailed classification with information on the the crops representing at least 10%
of the area.",
        "styleCode": "LCC",
        "extensions": [
          ".tif"
        ],
        "measureCaption": "Land Cover Classification",
        "measureUnit": "class",
        "slope": 1.0,
        "intercept": 0.0,
        "flags": null,
        "classes": {
          "1": {
            "caption": "Tree cover (closed)"
          },
          "4": {
            "caption": "Grassland"
          },
          "11": {
            "caption": "Vegetables"
          },
          "12": {
            "caption": "Fallow"
          },
          "18": {
            "caption": "Shrubland"
          },
          "19": {
            "caption": "Water"
          },
          "21": {
            "caption": "Unknown mixed crops"
          },
          "30": {
            "caption": "Non vegetation (reclass)"
```

```
          },
          "111": {
            "caption": "Vegetables irrigated"
          },
          "113": {
            "caption": "Orchard irrigated (closed)"
          },
          "121": {
            "caption": "Unknown mixed crops irrigated"
          },
          "122": {
            "caption": "Sugarcane irrigated"
          },
          "22.00": {
            "caption": "Sugarcane"
          }
        },
        "translate": [
          {
            "subDataset": null,
            "band": null,
            "format": "GTiff",
            "dataType": null,
            "noDataValue": null,
            "scale": null,
            "srs": null,
            "extent": null,
            "gridSubWindow": null,
            "srsSubWindow": null,
            "options": [
              "COMPRESS=LZW",
              "INTERLEAVE=BAND",
              "TILED=YES"
            ]
          }
        ],
        "additionalInfo": null,
        "tags": [
          "AWASH",
          "ETHIOPIA",
          "LAND"
        ],
        "links": [
          {
            "rel": "self",
            "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/AWA-LCC"
          }
        ]
      },
      {
        "workspaceCode": "ACME",
        "code": "MAP01",
        "caption": "My first Map",
        "description": "Land Cover Classification (Awash, Ethiopia - Dekadal)",
        "styleCode": "ACME-STYLE",
        "extensions": [
          ".tif"
        ],
        "measureCaption": "Land Cover Classification",
        "measureUnit": "class",
        "slope": 1.0,
```

```
        "intercept": 0.0,
        "flags": null,
        "classes": {
          "1": {
            "caption": "Tree cover (closed)"
          },
          "4": {
            "caption": "Grassland"
          },
          "11": {
            "caption": "Vegetables"
          },
          "12": {
            "caption": "Fallow"
          },
          "18": {
            "caption": "Shrubland"
          },
          "19": {
            "caption": "Water"
          },
          "21": {
            "caption": "Unknown mixed crops"
          },
          "30": {
            "caption": "Non vegetation (reclass)"
          },
          "111": {
            "caption": "Vegetables irrigated"
          },
          "113": {
            "caption": "Orchard irrigated (closed)"
          },
          "121": {
            "caption": "Unknown mixed crops irrigated"
          },
          "122": {
            "caption": "Sugarcane irrigated"
          },
          "22.00": {
            "caption": "Sugarcane"
          }
        },
        "translate": [
          {
            "subDataset": null,
            "band": null,
            "format": "GTiff",
            "dataType": null,
            "noDataValue": null,
            "scale": null,
            "srs": null,
            "extent": null,
            "gridSubWindow": null,
            "srsSubWindow": null,
            "options": [
              "COMPRESS=LZW",
              "INTERLEAVE=BAND",
              "TILED=YES"
            ]
          }
        ],
```

```
        "additionalInfo": null,
        "tags": [
          "AWASH",
          "LAND"
        ],
        "links": [
          {
            "rel": "self",
            "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/MAP01"
          }
        ]
      }
    ],
    "links": [
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps"
      }
    ]
  }
}
```


8.3. Get

Retrieves an instance of `map` resource.

Authentication/Authorization is not required

HTTP Request:

```
GET /catalog/workspaces/{w_code}/maps/{m_code}
```

Request Headers:

```
Accept: application/json
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the map belongs
m_code	the unique code of the map you want to retrieve.

Query Parameters:

The default query parameters apply.

Response Body:

If successful (HTTP response code = **200**), the response body contains the selected instance of `map` resource.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# get a selected workspace's map
curl -X GET -H "Accept: application/json"
"${BASE_URL}/catalog/workspaces/ACME/maps/AWA-LCC"

# RESPONSE:
{
  "requestId": "94f64119-fe5f-48c4-bfae-7d755f210ae4",
  "timestamp": 1670412992644,
  "runtime": 7,
  "status": 200,
  "message": "OK",
  "response": {
    "workspaceCode": "ACME",
    "code": "AWA-LCC",
    "caption": "Land Cover Classification (Awash, Ethiopia)",
    "description": "This land cover dataset at sub-national scale shows a detailed
classification with information on the the crops representing at least 10% of the
area.",
    "styleCode": "LCC",
    "extensions": [
      ".tif"
    ],
    "measureCaption": "Land Cover Classification",
    "measureUnit": "class",
    "slope": 1.0,
    "intercept": 0.0,
    "flags": null,
    "classes": {
      "1": {
        "caption": "Tree cover (closed)"
      },
      "4": {
        "caption": "Grassland"
      },
      "11": {
        "caption": "Vegetables"
      },
      "12": {
        "caption": "Fallow"
      },
      "18": {
        "caption": "Shrubland"
      },
```

```
      "19": {
        "caption": "Water"
      },
      "21": {
        "caption": "Unknown mixed crops"
      },
      "30": {
        "caption": "Non vegetation (reclass)"
      },
      "111": {
        "caption": "Vegetables irrigated"
      },
      "113": {
        "caption": "Orchard irrigated (closed)"
      },
      "121": {
        "caption": "Unknown mixed crops irrigated"
      },
      "122": {
        "caption": "Sugarcane irrigated"
      },
      "22.00": {
        "caption": "Sugarcane"
      }
    },
    "translate": [
      {
        "subDataset": null,
        "band": null,
        "format": "GTiff",
        "dataType": null,
        "noDataValue": null,
        "scale": null,
        "srs": null,
        "extent": null,
        "gridSubWindow": null,
        "srsSubWindow": null,
        "options": [
          "COMPRESS=LZW",
          "INTERLEAVE=BAND",
          "TILED=YES"
        ]
      }
    ],
    "additionalInfo": null,
    "tags": [
      "AWASH",
      "ETHIOPIA",
      "LAND"
    ],
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/AWA-LCC"
      },
      {
```

```
        "rel": "raster",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/AWA-LCC/raster"
      }
    ]
  }
}
```


8.4. Get (raster)

Raster JSON representation


```
{
  “workspaceCode”: string,
  “mapCode”: string,
  “code”: string,
  “caption”: string,
  “description”: string,
  “styleCode”: string,
  “measureCaption”: string,
  “measureUnit”: string,
  “slope”: double,
  “intercept”: double,
  “additionalInfo”: object,
  “tags”: [
    string
  ],
  “width”: integer,
  “height”: integer,
  “affineTransform”: [
    double
  ],
  “srs”: object,
  “extent”: object,
  “dataType”: string,
  “noDataValue”: double,
  “flags”: object,
  “classes”: object
}
```


FIELD	DESCRIPTION
workspaceCode	string: unique code of the workspace to which the resource

	belongs
mapCode	string: unique code of the map to which the resource is bound
code	string: unique code of the resource; assigned by the system using the following naming convention: <WORKSPACE_CODE>_<MAP_CODE>
caption	string: the caption of the map to which the resource is bound
description	string: the description of the map to which the resource is bound
styleCode	string: the style of the map to which the resource is bound
measureCaption	string: the measureCaption of the map to which the resource is bound
measureUnit	string: the measureUnit of the map to which the resource is bound
slope	double: the slope of the map to which the resource is bound
intercept	double: the intercept of the map to which the resource is bound
additionalInfo	object: the additionalInfo object of the map to which the resource is bound
tags	string: the tags array of the map to which the resource is bound
width	integer: the raster width expressed in pixels
height	integer: the raster height expressed in pixels

affineTransform	array: array of 6 elements (double)
	The affine transform (or geo transform) consists of six coefficients which map pixel/line (column/row) coordinates into georeferenced space using the following relationship:
	
	`Xgeo = GT(0) + X*GT(1) + Y*GT(2)`
	`Ygeo = GT(3) + X*GT(4) + Y*GT(5)`
	
	where:
	• Xgeo is the X coordinate expressed in the raster SRS units
	• Ygeo is the Y coordinate expressed in the raster SRS units
	• GT(n) is one of the six geo-transform coefficients (n=0..5)
	• X is the X coordinate of the pixel or column (0 indexed)
	• Y is the Y coordinate of the pixel or row (0 indexed)

	In case of north up images, the GT(2) and GT(4) coefficients are zero, and the GT(1) is pixel width, and GT(5) is pixel height. The (GT(0),GT(3)) position is the top left corner of the top left pixel of the raster. Note that the pixel/line coordinates in the above are from (0.0,0.0) at the top left corner of the top left pixel to at the bottom right corner of the bottom right pixel. The pixel/line location of the center of the top left pixel would therefore be (0.5,0.5).
srs	`object`: an object containing two properties that identify the raster SRS: • `proj4` - the SRS expressed as a PROJ4 declaration string • `epsg` - the SRS expressed as EPSG code. example: `{ "proj4": "+proj=utm +zone=37 +datum=WGS84 +units=m +no_defs", "epsg": "32637" }`
extent	`object`: an object containing two properties that identify the raster's extent: • `srs` - the extent expressed in the raster SRS coordinates • `wgs84` - the extent expressed in geographical coordinates example: `{ "srs": [ 518055, 912285, 613485, 988515 ], "wgs84": [ 8.253124164269858, 39.163942178982204, 8.941242294937885, 40.03226960732865 ] }`
dataType	`string`: specifies the data type of the raster's pixel values
noDataValue	`double`: specifies the value used to identify the pixels that do not contain actual/valid data.
flags	`object`: the flags object of the map to which the resource is

	bound
`classes`	`classes:` the classes object of the map to which the resource is bound

Retrieves the geospatial information of the raster bound to the map.

Authentication/Authorization is not required

HTTP Request:

```
GET /catalog/workspaces/{w_code}/maps/{m_code}/raster
```

Request Headers:

```
Accept: application/json
```

Path Parameters:

PARAMETER	DESCRIPTION
`w_code`	the unique code of the workspace to which the map belongs
`m_code`	the unique code of the map to which the raster is bound.

Query Parameters:

The default query parameters apply.

Response Body:

If successful (HTTP response code = **`200`**), the response body contains the raster instance bound to the selected map

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# get the raster info bound to a selected workspace's map
curl -X GET -H "Accept: application/json"
"${BASE_URL}/catalog/workspaces/ACME/maps/AWA-LCC/raster"
```

```
# RESPONSE:
{
  "requestId": "8b7bae15-01e0-42f0-a272-064cc2d57a28",
  "timestamp": 1670419267856,
  "runtime": 23,
  "status": 200,
  "message": "OK",
  "response": {
    "workspaceCode": "ACME",
    "mapCode": "AWA-LCC",
    "code": "ACME_AWA-LCC",
    "caption": "Land Cover Classification (Awash, Ethiopia)",
    "description": "This land cover dataset at sub-national scale shows a detailed
classification with information on the the crops representing at least 10% of the
area.",
    "styleCode": "LCC",
    "measureCaption": "Land Cover Classification",
    "measureUnit": "class",
    "slope": 1.0,
    "intercept": 0.0,
    "additionalInfo": null,
    "tags": [
      "AWASH",
      "ETHIOPIA",
      "LAND"
    ],
    "width": 3181,
    "height": 2541,
    "affineTransform": [
      518055.0,
      30.0,
      0.0,
      988515.0,
      0.0,
      -30.0
    ],
    "srs": {
      "proj4": "+proj=utm +zone=37 +datum=WGS84 +units=m +no_defs",
      "epsg": "32637"
    },
    "extent": {
      "srs": [
        518055.0,
        912285.0,
        613485.0,
        988515.0
      ],
      "wgs84": [
        8.253124164269858,
        39.163942178982204,
        8.941242294937885,
        40.03226960732865
      ]
    },
    "dataType": "Byte",
    "noDataValue": 255.0,
    "flags": null,
    "classes": {
      "1": {
        "caption": "Tree cover (closed)"
      },
```

```
      "4": {
        "caption": "Grassland"
      },
      "11": {
        "caption": "Vegetables"
      },
      "12": {
        "caption": "Fallow"
      },
      "18": {
        "caption": "Shrubland"
      },
      "19": {
        "caption": "Water"
      },
      "21": {
        "caption": "Unknown mixed crops"
      },
      "30": {
        "caption": "Non vegetation (reclass)"
      },
      "111": {
        "caption": "Vegetables irrigated"
      },
      "113": {
        "caption": "Orchard irrigated (closed)"
      },
      "121": {
        "caption": "Unknown mixed crops irrigated"
      },
      "122": {
        "caption": "Sugarcane irrigated"
      },
      "22.00": {
        "caption": "Sugarcane"
      }
    },
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/AWA-LCC"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/AWA-LCC/raster"
      },
      {
        "rel": "style",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC"
      },
      {
        "rel": "sld",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC/sld"
      },
      {
        "rel": "legend",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/styles/LCC/legend"
```

```
      }
    ]
  }
}
```

8.5. Update

To update an existing map, use the following PUT request and include the authorization described in Authorizing Requests.

HTTP Request:

```
PUT /catalog/workspaces/{w_code}/maps/{m_code}
```

Request Headers:

```
       Accept: application/json
 Content-Type: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the map belongs
m_code	the unique code of the map you want to update.

Response Body:

If successful (HTTP response code = **200**), the response body contains the updated instance of the selected map resource.

Example:

update some fields of the previously created MAP.json file and execute:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# update an existing map
curl -X PUT -H "Content-Type: application/json" -H "Accept: application/json" -H
```

```
"Authorization: Bearer ${ID_TOKEN}" -d @MAP.json
"${BASE_URL}/catalog/workspaces/ACME/maps"

# RESPONSE
{
  "requestId": "41152227-d147-4b4a-809c-f4a00d8e9bc0",
  "timestamp": 1670420621538,
  "runtime": 389,
  "status": 200,
  "message": "OK",
  "response": {
    "workspaceCode": "ACME",
    "code": "AWA-LCC",
    "caption": "Land Cover Classification (Awash, Ethiopia)",
    "description": "This land cover dataset at sub-national scale shows a detailed
classification with information on the the crops representing at least 10% of the
area.",
    "styleCode": "LCC",
    "extensions": [
      ".tif"
    ],
    "measureCaption": "Land Cover Classification",
    "measureUnit": "class",
    "slope": 1.0,
    "intercept": 0.0,
    "flags": null,
    "classes": {
      "1": {
        "caption": "Tree cover (closed)"
      },
      "4": {
        "caption": "Grassland"
      },
      "11": {
        "caption": "Vegetables"
      },
      "12": {
        "caption": "Fallow"
      },
      "18": {
        "caption": "Shrubland"
      },
      "19": {
        "caption": "Water"
      },
      "21": {
        "caption": "Unknown mixed crops"
      },
      "30": {
        "caption": "Non vegetation (reclass)"
      },
      "111": {
        "caption": "Vegetables irrigated"
      },
      "113": {
        "caption": "Orchard irrigated (closed)"
      },
      "121": {
        "caption": "Unknown mixed crops irrigated"
      },
      "122": {
        "caption": "Sugarcane irrigated"
```

```
      },
      "22.00": {
        "caption": "Sugarcane"
      }
    },
    "translate": [
      {
        "subDataset": null,
        "band": null,
        "format": "GTiff",
        "dataType": null,
        "noDataValue": null,
        "scale": null,
        "srs": null,
        "extent": null,
        "gridSubWindow": null,
        "srsSubWindow": null,
        "options": [
          "COMPRESS=LZW",
          "INTERLEAVE=BAND",
          "TILED=YES"
        ]
      }
    ],
    "additionalInfo": null,
    "tags": [
      "AWASH",
      "LAND"
    ],
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/AWA-LCC"
      },
      {
        "rel": "raster",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/workspaces/ACME/maps/AWA-LCC/raster"
      }
    ]
  }
}
```


8.6. Delete

To delete an existing map, use the following DELETE request and include the authorization described in Authorizing Requests.

HTTP Request:

```
DELETE /catalog/workspaces/{w_code}/maps/{m_code}
```

Request Headers:

```
      Accept: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
w_code	the unique code of the workspace to which the map belongs
m_code	the unique code of the map you want to delete.

Response Body:

If successful (HTTP response code = **200**), the response body contains the deleted instance of the selected map resource.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# delete an existing map
curl -X DELETE -H "Accept: application/json" -H "Authorization: Bearer ${ID_TOKEN}"
"${BASE_URL}/catalog/workspaces/ACME/maps/AWA-LCC"

# RESPONSE
{
  "requestId": "9017dde8-5f94-4184-be78-1a9976ca6a2a",
  "timestamp": 1670421038848,
  "runtime": 620,
  "status": 200,
  "message": "OK",
  "response": {
    "workspaceCode": "ACME",
    "code": "AWA-LCC",
    "caption": "Land Cover Classification (Awash, Ethiopia)",
    "description": "This land cover dataset at sub-national scale shows a detailed
classification with information on the the crops representing at least 10% of the
area.",
    "styleCode": "LCC",
    "extensions": [
      ".tif"
    ],
    "measureCaption": "Land Cover Classification",
```

```
    "measureUnit": "class",
    "slope": 1.0,
    "intercept": 0.0,
    "flags": null,
    "classes": {
      "1": {
        "caption": "Tree cover (closed)"
      },
      "4": {
        "caption": "Grassland"
      },
      "11": {
        "caption": "Vegetables"
      },
      "12": {
        "caption": "Fallow"
      },
      "18": {
        "caption": "Shrubland"
      },
      "19": {
        "caption": "Water"
      },
      "21": {
        "caption": "Unknown mixed crops"
      },
      "30": {
        "caption": "Non vegetation (reclass)"
      },
      "111": {
        "caption": "Vegetables irrigated"
      },
      "113": {
        "caption": "Orchard irrigated (closed)"
      },
      "121": {
        "caption": "Unknown mixed crops irrigated"
      },
      "122": {
        "caption": "Sugarcane irrigated"
      },
      "22.00": {
        "caption": "Sugarcane"
      }
    },
    "translate": [
      {
        "subDataset": null,
        "band": null,
        "format": "GTiff",
        "dataType": null,
        "noDataValue": null,
        "scale": null,
        "srs": null,
        "extent": null,
        "gridSubWindow": null,
        "srsSubWindow": null,
        "options": [
          "COMPRESS=LZW",
          "INTERLEAVE=BAND",
          "TILED=YES"
        ]
```

```
      }
    ],
    "additionalInfo": null,
    "tags": [
      "AWASH",
      "LAND"
    ]
  }
}
```

9. Storage User

Storage Users can access the DATA bucket of all workspaces present in the system. Similarly to system Users, GISMGR manages resource access permissions using the concept of Users and Roles. A Storage User is identified by a valid and existing email address. A Role specifies which operations that user can perform on resources belonging to a specific workspace.

USER JSON representation

```
{
  "email": string,
  "role": string
}
```

FIELD	DESCRIPTION
email	string: the email address that identifies the user. cannot be null; must be a valid and existing email address.
role	string: the role assigned to the user within all workspaces' DATA buckets cannot be null; allowed values are: • EDITOR • VIEWER

9.1. Roles

ROLE	DESCRIPTION
EDITOR	• can read and edit bucket metadata, including IAM policies. • can create, list, view, download, replace, edit and delete objects • can read object metadata when listing (excluding IAM policies); • can create, delete, and list tag bindings
VIEWER	• can list a bucket's contents and read bucket metadata, excluding IAM policies. • can read/download objects and their metadata, excluding ACLs. • can read object metadata when listing objects (excluding IAM policies).

9.2. Create

To create a storage user, use the following POST request and include the authorization described in Authorizing Requests.

ATTENTION: only System Administrators can manage storage users

HTTP Request:

```
POST /catalog/system/storage/users
```

Request Headers:

```
       Accept: application/json
 Content-Type: application/json
Authorization: Bearer <ID_TOKEN>
```

Request Body:

The request body is an instance of `storage user`.

Response Body:

If successful (HTTP response code = **201**), the response body contains a newly created instance of `storage user`.

Example:

prepare a `STORAGE.USER.json` file with the following content, set a valid email address:

```
{
  “email”: “elmer.fudd@looney-tunes.com”,
  “role”: “EDITOR”
}
```

execute the request:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# add a new storage user
curl -X POST -H "Content-Type: application/json" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -d @STORAGE.USER.json
```

```
"${BASE_URL}/catalog/system/storage/users"

# RESPONSE
{
  "requestId": "5cc50dc7-ee1b-47a8-8f0e-8dab1f53b5e9",
  "timestamp": 1670428533633,
  "runtime": 1969,
  "status": 201,
  "message": "Created",
  "response": {
    "email": "elmer.fudd@looney-tunes.com",
    "role": "EDITOR",
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/system/storage/users"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/system/storage/users/elmer.fudd@loon
ey-tunes.com"
      }
    ]
  }
}
```

9.3. List

Retrieves a list of `storage user` resources. The request must include the bearer ID token.

ATTENTION: only System Administrators can manage storage users

HTTP Request:

```
GET /catalog/system/storage/users
```

Request Headers:

```
       Accept: application/json
Authorization: Bearer <ID_TOKEN>
```

Query Parameters:

PARAMETER	DESCRIPTION
`asMap`	boolean; optional (default=false) returns the result as an object instead as an array of `storage user` resources

Response Body:

If successful (HTTP response code = **`200`**), the response body contains an array of `storage user` resource items. If the `asMap=true` query parameter has been specified, the response body contains an object where the key/value pairs represent the user email address and its assigned role.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# retrieves the storage users list (as a map)
curl -X GET -H "Accept: application/json" -H "Authorization: Bearer ${ID_TOKEN}"
"${BASE_URL}/catalog/system/storage/users?asMap=true"

@ RESPONSE
{
  "requestId": "15d3df50-addb-4622-8c18-514647cc47db",
  "timestamp": 1670429334124,
  "runtime": 15,
  "status": 200,
  "message": "OK",
  "response": {
    "yosemite.sam@looney-tunes.com": "VIEWER",
    "elmer.fudd@looney-tunes.com": "EDITOR",
    "sylvester@gmail.com": "EDITOR"
  }
}
```

9.4. Get

Retrieves an existing `storage user` resource item. The request must include the bearer ID token.

ATTENTION: only System Administrators can manage storage users

HTTP Request:

```
GET /catalog/system/storage/users/{email}
```

Request Headers:

```
       Accept: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
email	the email address that identifies the user/account

Response Body:

If successful (HTTP response code = **200**), the response body contains the selected instance of `storage user` resource.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# get an existing storage user
curl -X GET -H "Accept: application/json" -H "Authorization: Bearer ${ID_TOKEN}"
"${BASE_URL}/catalog/system/storage/users/elmer.fudd@looney-tunes.com"

# RESPONSE
{
  "requestId": "5e8d6f00-e9d5-484d-9aa1-ba12c180a23f",
  "timestamp": 1670430157026,
  "runtime": 19,
  "status": 200,
  "message": "OK",
  "response": {
    "email": "elmer.fudd@looney-tunes.com",
    "role": "VIEWER",
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/system/storage/users"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/system/storage/users/elmer.fudd@loon
```

```
ey-tunes.com"
      }
    ]
  }
}
```

9.5. Update

To update an existing storage user, use the following PUT request and include the authorization described in Authorizing Requests.

ATTENTION: only System Administrators can manage storage users

HTTP Request:

```
PUT /catalog/system/storage/users/{email}
```

Request Headers:

```
       Accept: application/json
 Content-Type: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
email	the email address that identifies the user/account

Request Body:

The request body is an instance of `storage user`.

Response Body:

If successful (HTTP response code = **200**), the response body contains a the updated instance of `storage user` resouce.

Example:

update the previously created `STORAGE.USER.json` file changing the user role from `EDITOR` to `VIEWER`, then execute the request:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# update an existing storage user
curl -X PUT -H "Content-Type: application/json" -H "Accept: application/json" -H
"Authorization: Bearer ${ID_TOKEN}" -d @STORAGE.USER.json
"${BASE_URL}/catalog/system/storage/users/elmer.fudd@looney-tunes.com"

# RESPONSE
{
  "requestId": "27f7e28a-cb2c-45d2-8836-f24e18168aa8",
  "timestamp": 1670430625852,
  "runtime": 12,
  "status": 200,
  "message": "OK",
  "response": {
    "email": "elmer.fudd@looney-tunes.com",
    "role": "VIEWER",
    "links": [
      {
        "rel": "parent",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/system/storage/users"
      },
      {
        "rel": "self",
        "href":
"http://data.apps.fao.org/gismgr/api/v2/catalog/system/storage/users/elmer.fudd@loon
ey-tunes.com"
      }
    ]
  }
}
```

9.6. Delete

To delete an existing storage user, use the following DELETE request and include the authorization described in Authorizing Requests.

ATTENTION: only System Administrators can manage storage users

HTTP Request:

```
DELETE /catalog/system/storage/users/{email}
```

Request Headers:

```
      Accept: application/json
Authorization: Bearer <ID_TOKEN>
```

Path Parameters:

PARAMETER	DESCRIPTION
`email`	the email address that identifies the user/account

Response Body:

If successful (HTTP response code = **`200`**), the response body contains the deleted instance of `storage user` resource.

Example:

```
# GISMGR base URL
BASE_URL=https://data.apps.fao.org/gismgr/api/v2

# previously obtained refresh token
ID_TOKEN=5ea2f6941e8d02566c38d4cb2115f38...

# delete an existing storage user
curl -X DELETE -H "Accept: application/json" -H "Authorization: Bearer ${ID_TOKEN}"
"${BASE_URL}/catalog/system/storage/users/elmer.fudd@looney-tunes.com"

# RESPONSE
{
  "requestId": "9d6d424b-54c0-40c1-903c-db15d1dcabe9",
  "timestamp": 1670431001847,
  "runtime": 1905,
  "status": 200,
  "message": "OK",
  "response": {
    "email": "elmer.fudd@looney-tunes.com",
    "role": "VIEWER"
  }
}
```

THIS IS THE LAST PAGE OF THE DOCUMENT