EMMi APIs

REST v.1 — Last update: 21 September 2022

Grano

Table of Contents

1. Introduction

EMMi REST API is public API for integrating EMMi Media Bank to third party applications.

It provides some basic operations for fetching folders, assets and files stored in EMMi.

REST API can be enabled for any EMMi instance (versions 5.5 and greater) at request. Swagger documentation for REST API is available. When enabled there is Swagger UI in following path at EMMi instance's web site:
`/papi/v1/swagger/ui/index`
(for example https://grano.emmi.fi/papi/v1/swagger/ui/index)

2. Concepts of EMMi Media Bank

2.1. Asset and files

Asset is an object which describes the digital asset with related metadata and files.

Asset itself does not define where or how the file exists physically, it only defines the metadata and how the file is to be referenced. Each asset has an unique identifier.

Assets can be read using the methods in REST API, but modifying or adding assets is not currently possible.

Asset is always linked to one or more folders. User access permissions for the asset are inherited from these linked relations.

Each asset has the following properties:

- Name
- Status
- Description
- Publish time

Optional properties for assets include:

- Links to property field values
- Links to keywords

Files

Asset can contain one master file and starting from EMMi version 6.0 multiple attachment files can be linked to asset using different attachment types.

Each file can have multiple file versions of which only one is considered to be an active version. Only active versions are accessible using REST API. Each file version has an unique identifier. File version includes information about the type, name, etc. of the file. File version includes links to available conversions.

2.2. Conversion

Conversions are download options for files.

There are three different kinds of possible conversions:

- Original. Usually there is only one of these and it refers to the original file of the file version.
- Automatic conversion. Conversion which is created on request from the original file version.
    - Automatic conversion will be generated after the first request for some specific conversion of a file. The REST API download operation will return HTTP response code 202 (conversion started) or 204 (conversion waiting or processing) when the conversion is not yet available. The same request can be repeated until the the requested conversion is ready and the file binary downloads with response code 200. Some decent delay (for example 5 seconds) between requests is recommended. Conversions are cached after creation and following requests for the same conversion will return the file binary immediately.
- Manual conversion. Separate file which has been manually attached as an conversion.

Available conversions cannot be manipulated using REST API. They are managed using EMMi user interface with admin permissions.

Document Conversions Powered By Aspose

2.3. Localized Text

Majority of the text information in the system is saved in multilingual form. The text can be the same for all languages or different languages can have different texts.

In REST API these text values are returned as JSON objects where the keys are language codes. If the language for value is not specified the language code is “x-default”

Example JSON for non-translated text value:

```
"Name": {
  "x-default": "This is name"
}
```

Example JSON for translated text value:

```
"Name": {
  "en": "This is name",
  "fi": "Tämä on nimi",
  "sv": "Detta är namnet",
  "ru": "Это имя"
}
```

2.4. Property Fields

The system can have user defined amount of property fields of which there are several types available:

- single line text
- multi line text
- formatted text (HTML)
- radio button list
- boolean
- multi select list
- drop down list
- date and time
- date
- time
- number
- checkbox list

Some types are kind of option (radio button list, multi select list, drop down list, checkbox list) which has a predefined array of values to use. Others are defined during input.

Values of property fields are linked to elements.

Using REST API it is possible to fetch information about available property fields and field values. Currently it is not possible to modify property fields.

2.5. Keywords

The system can have a hierarchical group of keywords which can be linked to assets. The depth of hierarchy or the amount of keywords is not restricted.

Example keyword hierarchy:

- Food
  - Soup
    - Pea soup
    - Vegetable soup
    - Beef soup
  - Bread
    - Toast
    - Rye bread
    - Roll

Hierarchical keywords inherits all of the parent keywords. For example: if we have an asset which is linked to keyword “Beef soup”, it would also be found when searching for “Soup” or “Food”.

It is not possible to browse or modify keyword hierarchies using the REST API. Keywords linked to assets are returned with assets and keywords can be used for searching assets.

2.6. Folder Hierarchy

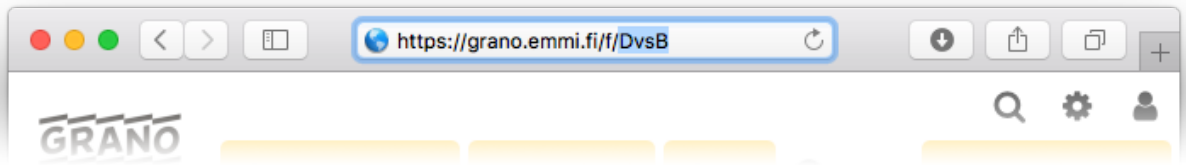
Assets in EMMi are categorised in a folder hierarchy. The amount of folders or depth of hierarchy is not restricted. Single asset can be linked to multiple folders. Each folder has an unique identifier which can be used to retrieve subfolders and files linked to it.

Example of folder hierarchy:

- Machinery (root folder: level 0)
  - Cars (subfolder: level 1)
    - Logos (subfolder: level 2)
    - Pictures (subfolder: level 2)
  - Heavy Machinery (subfolder: level 1)
    - Logos (subfolder: level 2)
    - Pictures (subfolder: level 2)

Folder hierarchy can be retrieved using REST API, but currently it cannot be modified.

In EMMi UI the folder id can be located at browser's address bar:



2.7. User Groups and Access Permission

The access permissions for EMMi service are controlled by user groups, folders and services.

Login to the service requires username and password. Usernames are unique in system and they relate to user account. Single user account can belong to multiple user groups.

Same username and password can be used to authenticate using REST API and user interface.

Access privileges in EMMi are defined by user groups linked to folders. User groups can be linked with different access permissions:

- read
- modify assets
- add assets
- remove assets
- add subfolders
- modify folder

It is not currently possible to modify user groups or access permissions using REST API.

2.8. Search options

SearchField options:

- Any (any text field, type: text)
- Name (type: text)
- Description (type: text)
- Created (type: unixdatetime)
- PublishTime (type: unixdatetime)
- StatusId (type: integer)
- KeyWord (type: text)
- FolderId (Search options(optional): IncludeSubfolders)
- Modified (type: unixdatetime)
- Extension (type: text)
- FileName (type: text)
- FileSize (type: integer)
- property field identifier (type: PropertyFieldType)]

SearchOption options:

- General (Default valid search option for text type fields)
- Contains (Valid for text type fields)
- ContainsWords (Valid for text type fields)
- Exactly (Valid for all search fields. default search option for integer and unixdatetime type fields)
- StartsWith (Valid for text type fields)
- EndsWith (Valid for text type fields)
- Regex (Valid for text type fields)
- BetweenRange (Valid for integer type,unixdatetime type fields)
- SmallerThan (Valid for integer type,unixdatetime type fields)
- GreaterThan (Valid for integer type,unixdatetime type fields)
- IncludeSubfolders (Valid only for FolderId search)

3. Example Use Case

3.1. Media Bank

3.1.1. Authentication

Request: POST /papi/v1/auth/Login

Provide valid username and password and send the login request

Parameter	Parameter Type	Mandatory	Description
User info	Body	yes	User and Password in JSON format in the request body. See the format below

```
{
  "User": "string",
  "Password": "string"
}
```

Response as a valid response the “x-auth-token” is returned.

```
{
    "x-auth-token": "[Token]"
}
```

The authentication token must be provided as a x-auth-token header for every api request except login request.

> In Swagger UI x-auth-token value can be used as ‘api_key’.

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/auth/Login' \
--header 'Content-Type: application/json' \
--data-raw '{"User": "[User]",  "Password": "[Password]"}'
```

3.1.2. Search asset

Request: POST /papi/v1/asset/search

Parameter	Parameter Type	Mandatory	Description
Search info	Body	yes	Search query in JSON format, see examples below
x-auth-token	Header	yes	The authentication token must be provided as a x-auth-token header for every api request. More information about authentication token.

Query parameters:

Query Parameters	Parameter Type	Mandatory	Description
SearchField	string	yes	Field to search from
SearchOption	string	no	Search option behavior
ValuesList	array	yes	Value or values to search
Exclude	boolean	no	Exclude the SearchField

More information about search options: “Search Options”

Examples

Folder Search

The below example is search the assets in the folder “gmXs” and its subfolders.

```
[
 {
   "SearchField": "FolderId",
   "SearchOption": "IncludeSubfolders",
   "ValuesList": [
     "gmXs"
   ]
 }
]
```

The below example is search the assets in the folder “gmXs” only.

```
[
  {
    "SearchField": "FolderId",
    "ValuesList": [
      "gmXs"
    ]
  }
]
```

The below example is search the all assets excluding the asset in the folder “gmXs” and its subfolders.

```
[
  {
    "SearchField": "FolderId",
    "SearchOption": "IncludeSubfolders",
    "ValuesList": [
      "gmXs"
    ],
    "Exclude": true
  }
]
```

Name search

Search all asset where asset name contains the letters ‘test’.

```
[
  {
    "SearchField": "Name",
    "SearchOption": "contains",
    "ValuesList": [
      "test"
    ]
  }
]
```

Search all asset where asset name = “SGR28815”

```
[
  {
    "SearchField": "Name",
    "SearchOption": "Exactly",
    "ValuesList": [
      "SGR28815"
    ]
```

```
  }
]
```

Search all asset where asset name contains the word “image”.

```
 [
  {
    "SearchField": "Name",
    "SearchOption": "containwords",
    "ValuesList": [
      "image"
    ]
  }
]
```

Combined search

Search queries can be combined putting the queries in an array.

example searching from folder id “gmXs” and its subfolders with name field containing “test”

```
 [
  {
    "SearchField": "FolderId",
    "SearchOption": "IncludeSubfolders",
    "ValuesList": [
      "gmXs"
    ]
  },
  {
    "SearchField": "Name",
    "SearchOption": "contains",
    "ValuesList": [
      "test"
    ]
  }
]
```

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/asset/search'
\
--header 'x-auth-token: [Session token]' \
--header 'Content-Type: application/json' \
--data-raw ' [
```

```
  {
    "SearchField": "FolderId",
    "SearchOption": "IncludeSubfolders",
    "ValuesList": [
      "gmXs"
    ]
  }
]'
```

3.1.3. Download file

Request : GET /papi/v1/file/download/[File id]/[Conversion id]

Downloads a file from an asset using a conversion.

Parameter	Parameter Type	Mandatory	Description
File id	Path	yes	File id is a file unique identifier.In order to get the file id use search request. See examples : Search asset
Conversion id	Path	yes	Conversion identifer. • If conversion id = original conversion id or preview files conversion id,then the file is immediately available to download. • If conversion id is other than original conversion id or preview files conversion id and the file with requested conversion is not available, then the file has to be generated by the system before download. • Available conversion id's for the file can be found at **Asset.MasterFile.Conversions** or **Asset.AttachedFiles.FileInfo.Conversions**. More information about conversions
x-auth-token	Header	yes	The authentication token must be provided as a x-auth-token header for every api download request. More information about authentication token.

Example of the asset object with master file id and conversions:

```
...
        "MasterFile": {
        "Id": "1065",
        "Filename": "image",
        "Extension": "png",
        "Created": 1597836932,
        "Size": 507224,
        "PreviewSupported": "Ready",
        "Previews": [
            "1",
            "3"
        ],
        "Conversions": [
            "1",
            "3",
```

```
            "6"
        ],
...
```

If the requested conversion is immediately available (like original, id=1) the HTTP response with status code 200 contains the file binary.
If the conversion has to be generated before downloading the HTTP response then one of the following status code (other than 200) is returned as shown is below table:

Response code	Response details	Description
200	Ok	The file with requested conversion is available and the file binary is retruned in HTTP response.
202	Conversion started.	The file with requested conversion is not available yet.The file conversion process has started. In this senario the download request has to be repeated after decent delay(few seconds) until the conversion completed.
204	Conversion Waiting. / Conversion Processing.	The file with requested conversion is not available yet.The file conversion is either in process or in the queue. In this senario the download request has to be repeated after decent delay(few seconds) until the conversion completed.
500	Conversion failed.	File conversion failed for the requested conversion.
403	File version /Conversion setting does not exist or is inaccessible.	Either requested file id/conversion id is wrong or the user don’t have permissions for the requested file.

cURL example

```
curl --location --request GET 'https://[EMMi instance]/papi/v1/file/downloa
d/[File id]/[Conversion id]' \
--header 'x-auth-token: [Session token]'
```

3.1.4. Upload file to folder

Request : POST /papi/v1/folder/[Folder id]/upload/[File name]

Uploads a file to a folder. This action will create a new asset in the folder.

Parameter	Parameter Type	Mandatory	Description
Folder id	Path	yes	Unique folder id where to upload the file. More information about folders
File name	Path	yes	filename containing the extension.
File content	Body	yes	Request body containing the file as a binary file.
x-auth-token	Header	yes	The authentication token must be provided as a x-auth-token header for every api request. More information about authentication token.

Response: JSON – Asset object.
The new asset JSON path is: *$.Id*

```
{
    "Id": "2179",
    "Name": {
        "x-default": "image"
...
```

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/folder/[Folde
r id]/upload/image.png' \
--header 'x-auth-token: [Session token]' \
--header 'Content-Type: image/png' \
--data-binary '@/Downloads/image.png'
```

3.1.5. Upload asset master file

Request : POST /papi/v1/asset/[Asset id]/upload/[File name]

Uploads a file to a new version of the asset's master file. This operation does not remove the existing file, it creates a new version.
The past versions of the files can be only modified and removed from the UI.

Parameter	Parameter Type	Mandatory	Description
Asset id	Path	yes	Unique asset id to upload the new master file.
File name	Path	yes	Filename containing the extension.
File content	Body	yes	Request body containing the file as a binary file.
x-auth-token	Header	yes	The authentication token must be provided as a x-auth-token header for every api request. More information about authentication token.

Response: JSON – Asset object.
The new masterfile id's JSON path is: *$.MasterFile.Id*

```
...
        "MasterFile": {
             "Id": "1061",
             "Filename": "image",
...
```

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/asset/[Asset id]/upload/image.png' \
--header 'x-auth-token: [Session token]' \
--header 'Content-Type: image/png' \
--data-binary '@/Downloads/image.png'
```

3.1.6. Upload file version

Request : POST /papi/v1/file/upload/[File id]/[File name]

Uploads a new file version to the existing asset attachment or a master file. This operation does not remove the existing file, it creates a new version.
The past versions of the files can be only modified and removed from the UI.

Parameter	Parameter Type	Mandatory	Description
File id	Path	yes	Unique file id to upload the new version.
File name	Path	yes	Filename containing the extension.
File content	Body	yes	Request body containing the file as a binary file.
x-auth-token	Header	yes	The authentication token must be provided as a x-auth-token header for every api request. More information about authentication token.

Response: JSON – Asset object.
Master file id can be found from path *$.MasterFile.Id*

```
...
        "MasterFile": {
             "Id": "1061",
             "Filename": "image",
...
```

Attachment id(s) can be found from path *$.AttachedFiles..FileInfo..Id*

```
...
    "AttachedFiles": [
            {
                "AttachmentType": "imagePNG",
                "FileInfo": {
                    "Id": "1070",
...
```

cURL example

```
curl --location --request POST 'https://EMMi instance]/papi/v1/file/upload/[Fi
le id]/image.png' \
--header 'x-auth-token: [Session token]' \
```

```
--header 'Content-Type: image/png' \
--data-binary '@/Downloads/robot.png'
```

use “GET /papi/v1/asset/..” api and get asset file Ids.

3.1.7. Upload asset attachment file

Request: POST /papi/v1/asset/[Asset id]/attachment/[Attachment id]/upload/[File name]

Uploads a file to an asset as an attachment. Attachment id is mandatory and can be accessed via UI or attachment type API command.
There is two ways defining the attachment ids, using system generated (integer) or user given (string).

Parameter	Parameter Type	Mandatory	Description
Asset id	Path	yes	Unique asset id to upload the new version.
Attachment id	Path	yes	• System given id (integer) • User given key (string)
File name	Path	yes	Filename containing the extension.
File content	Body	yes	Request body containing the file as a binary file.
x-auth-token	Header	yes	The authentication token must be provided as a x-auth-token header for every api request. More information about authentication token.

Response: JSON – Asset object.
Attachment id(s) can be found from path *$.AttachedFiles..FileInfo..Id*

```
...
    "AttachedFiles": [
            {
                "AttachmentType": "imagePNG",
                "FileInfo": {
                    "Id": "1070",
...
```

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/asset/[Asset i
d]/attachment/[Attachment id]/upload/image.png' \
--header 'x-auth-token: [Session token]' \
--header 'Content-Type: image/png' \
--data-binary '@Downloads/robot1.png'
```

Tips:

To get the available attachments and ids, use the attachment type command.

Request: GET /papi/v1/asset/attachmenttype

Parameter	Parameter Type	Mandatory	Description
x-auth-token	Header	yes	The authentication token must be provided as a x-auth-token header for every api request. More information about authentication token.

Response: JSON – Attachment details.
Attacment id's JSON path is: *$..Identifier*

```
[
    {
        "Identifier": "photoshopFile",
        "Name": {
            "x-default": "Photoshop File"
        },
...
]
```

3.1.8. Save Asset

More information about save/edit/delete property field values: ‘Save Property Field Values’.

Create new asset

“LinkedFolders” are mandatory for new asset. “PropertyValues” are optional.

```
{
    "Name": "asset name", // saving name text as default value
    "Description": {
        "fi": "asset Finnish description" // saving description for Finnish la
nguage
    },
    "LinkedFolders": [
        "bKx2"
    ],
    "PropertyValues": [
        {
            "FieldIdentifier": "TextPropertyFieldKey",
            "Value": "PropertyFieldValue"
        },
        {
            "FieldIdentifier": "CheckBoxKey",
            "Value": "CheckBox1_Option1"
        }
    ]
}
```

Save existing asset

Below is the example to update the asset name and adding one more number type property field to the existing asset.
if Id = 0 or not defined then asset is saved as new asset.

```
{
    "Id": "asset Key",
    "Name": {
        "en": "updatedName",
        "fi": "FinnishText"
    },
    "PropertyValues": [
        {
```

```
            "FieldIdentifier": "NumberTypePropertyFieldKey",
            "Value": 5
        }
    ]
}
```

3.1.9. Save Folder

More information about save/edit/delete property field values: ‘Save Property Field Values’.

Create new folder

If ParentId = 0 or not defined then the folder is saved to root level.
“FolderRights” and “PropertyValues” are optional.

```
{
    "ParentId": "ParentFolderKey",
    "Name": "Folder Name",
    "Description": "Folder Desc",
    "PropertyValues": [
        {
            "FieldIdentifier": "TextPropertyFieldKey",
            "Value": "PropertyFieldValue"
        },
        {
            "FieldIdentifier": "CheckBoxKey",
            "Value": "CheckBox1_Option1"
        }
    ],
    "FolderRights": [
        {
            "AllRights": true,
            "UserGroupId": "1"
        },
        {
            "AddAssets": true,
            "EditAssets": true,
            "DeleteAssets": true,
            "UserGroupId": 5
        }
    ]
}
```

Save existing folder

Below is the example to update the folder name and adding one more number type property field to the existing folder.
if Id = 0 or not defined then folder is saved as new folder.

```
{
    "Id": "Folder Key",
    "Name": {
        "en": "updatedName",
        "fi": "FinnishText"
    },
    "PropertyValues": [
        {
            "FieldIdentifier": "NumberTypePropertyFieldKey",
            "Value": 5
        }
    ]
}
```

Move existing folder

Below is the example to move existing folder(and sub folders) under other folder.

```
{
    "Id":"Folder Key",
    "ParentId": "Folder key of parent folder"
}
```

3.1.10. Save Property Field Values

If no value is assigned to “Value”, then property field value will be removed/deleted.

The user can add/edit/delete single or multiple property field values.

Save text/html/multilinetext type property field value

1. First text type property field “TextPropertyFieldKey1” will save the text for English and Finnish.
2. Second text type property field “TextPropertyFieldKey2” will save the default text.

```
"PropertyValues": [
{
        "FieldIdentifier": "TextPropertyFieldKey1",
        "Value":{"en":"English text", "fi":"Finnish text"    }
},
{
        "FieldIdentifier": "TextPropertyFieldKey2",
        "Value": "default text"
}
]
```

Edit or delete text/html/multilinetext type property field value

1. First text type property field “TextPropertyFieldKey1” will save the text for Swedish language and update existing the Finnish text. The save action will not delete the existing English text and final text for “TextPropertyFieldKey1” = {“en”:“English text”, “sv”:“Swedish text”, “fi”:“Updated Finnish text” }
2. Second text type property field “TextPropertyFieldKey2” will be remove or delete, as no value is assigned to “Value”.

```
"PropertyValues": [
{
        "FieldIdentifier": "TextPropertyFieldKey1",
        "Value": {        "sv":"Swedish text", "fi":"Updated Finnish text"   }
},
{
        "FieldIdentifier": "TextPropertyFieldKey2"
}
]
```

Save Boolean property field

1. First boolean type property field “BooleanPropertyFieldKey1” will save value = false.
2. Second boolean type property field “BooleanPropertyFieldKey2” will save value = true.

```
"PropertyValues": [
{
        "FieldIdentifier": "BooleanPropertyFieldKey1",
        "Value": 0
},
{
        "FieldIdentifier": "BooleanPropertyFieldKey2",
        "Value": 1
}
]
```

Edit or delete Boolean property field

1. First boolean type property field “BooleanPropertyFieldKey1” will save value = true.
2. Second boolean type property field “BooleanPropertyFieldKey2” will be remove or delete, as no value is assigned to “Value”.

```
"PropertyValues": [
{
        "FieldIdentifier": "BooleanPropertyFieldKey1",
        "Value": 1
},
{
        "FieldIdentifier": "BooleanPropertyFieldKey2"
}
]
```

Save Number property field

1. First number type property field “NumberPropertyFieldKey1” will save value = 20.
2. Second number type property field “NumberPropertyFieldKey2” will save value = -70.

```
"PropertyValues": [
{
        "FieldIdentifier": "NumberPropertyFieldKey1",
        "Value": 20
},
{
        "FieldIdentifier": "NumberPropertyFieldKey2",
        "Value": -70
}
]
```

Edit or delete Number property field

1. First number type property field “NumberPropertyFieldKey1” will save value = 50.
2. Second number type property field “NumberPropertyFieldKey2” will be remove or delete, as no value is assigned to “Value”.

```
"PropertyValues": [
{
        "FieldIdentifier": "NumberPropertyFieldKey1",
        "Value": 50
},
{
        "FieldIdentifier": "NumberPropertyFieldKey2"
}
]
```

Save Radio button and Drop Down option property field

1. First Radio type property field “RadioPropertyFieldKey1” will save second option “RadioPropertyFieldKey1_optionKey2”.
2. Second Radio type property field “RadioPropertyFieldKey2” will save forth option “RadioPropertyFieldKey2_optionKey4”.
3. Third Radio type property field “RadioPropertyFieldKey3” will save third option “RadioPropertyFieldKey3_optionKey3”.
4. First DropDown type property field “DropDownPropertyFieldKey1” will save second option “DropDownPropertyFieldKey1_optionKey2”.
5. Second Dropdown type property field “DropDownPropertyFieldKey2” will save third option “DropDownPropertyFieldKey1_optionKey3”.

```
"PropertyValues": [
{
        "FieldIdentifier": "RadioPropertyFieldKey1",
        "Value": "RadioPropertyFieldKey1_optionKey2"
},
{
        "FieldIdentifier": "RadioPropertyFieldKey2",
        "Value": "RadioPropertyFieldKey2_optionKey4"
},
{
        "FieldIdentifier": "RadioPropertyFieldKey3",
        "Value": "RadioPropertyFieldKey2_optionKey3"
},
{
        "FieldIdentifier": "DropDownPropertyFieldKey1",
        "Value": "DropDownPropertyFieldKey1_optionKey2"
},
```

```
{
        "FieldIdentifier": "DropDownPropertyFieldKey2",
        "Value": "DropDownPropertyFieldKey2_optionKey3"
}
]
```

Edit or delete Radio button or Drop Down option property field

1. First Radio type property field “RadioPropertyFieldKey1” will save sixth option “RadioPropertyFieldKey1_optionKey6”.
2. Second Radio type property field “RadioPropertyFieldKey2” will save first option “RadioPropertyFieldKey2_optionKey1”.
3. Third Radio type property field “RadioPropertyFieldKey3” will be remove or delete, as no value is assigned to “Value”.
4. First DropDown type property field “DropDownPropertyFieldKey1” will save forth option “DropDownPropertyFieldKey1_optionKey4”.
5. Second DropDown type property field “DropDownPropertyFieldKey2” will be remove or delete, as no value is assigned to “Value”.

```
"PropertyValues": [
{
        "FieldIdentifier": "RadioPropertyFieldKey1",
        "Value": "RadioPropertyFieldKey1_option6"
},
{
        "FieldIdentifier": "RadioPropertyFieldKey2",
        "Value": "RadioPropertyFieldKey2_option1"
},
{
        "FieldIdentifier": "RadioPropertyFieldKey3"
},
{
        "FieldIdentifier": "DropDownPropertyFieldKey1",
        "Value": "DropDownPropertyFieldKey1_optionKey4"
},
{
        "FieldIdentifier": "DropDownPropertyFieldKey2"
}
]
```

Save Checkbox button and MultiSelectList option property field

1. Checkbox type property field “CheckboxPropertyFieldKey1” will save first, third and sixth options “CheckboxPropertyFieldKey1_optionKey1”, “CheckboxPropertyFieldKey1_optionKey3”, “CheckboxPropertyFieldKey1_optionKey6”.
2. MultiSelectList type property field “MultiSelectListPropertyFieldKey5” will save second and forth

option “MultiSelectListPropertyFieldKey5_optionKey2”, “MultiSelectListPropertyFieldKey5_optionKey4”.

```
"PropertyValues": [
{
        "FieldIdentifier": "CheckboxPropertyFieldKey1",
        "Value": "CheckboxPropertyFieldKey1_optionKey1"
},
{
        "FieldIdentifier": "CheckboxPropertyFieldKey1",
        "Value": "CheckboxPropertyFieldKey1_optionKey3"
},
{
        "FieldIdentifier": "CheckboxPropertyFieldKey1",
        "Value": "CheckboxPropertyFieldKey1_optionKey6"
},
{
        "FieldIdentifier": "MultiSelectListPropertyFieldKey5",
        "Value": "MultiSelectListPropertyFieldKey5_optionKey2"
},
{
        "FieldIdentifier": "MultiSelectListPropertyFieldKey5",
        "Value": "MultiSelectListPropertyFieldKey5_optionKey4"
}
]
```

Edit Checkbox button and delete MultiSelectList option property field

1. Checkbox type property field “CheckboxPropertyFieldKey1” will save forth option only “CheckboxPropertyFieldKey1_optionKey4”.
2. MultiSelectList type property field “MultiSelectListPropertyFieldKey5” will be remove or delete, as no value is assigned to “Value”.

```
"PropertyValues": [
{
        "FieldIdentifier": "CheckboxPropertyFieldKey1",
        "Value": "CheckboxPropertyFieldKey1_optionKey4"
},
{
        "FieldIdentifier": "MultiSelectListPropertyFieldKey5"
}
]
```

3.2. Project Management

3.2.1. Create & Get Project

Create or Modify Project

Request : POST /papi/v1/project

Save a new project or edit an existing one. If the project Id = 0 or not defined then the project is saved as a new.

Parameter	Parameter Type	Mandatory	Description
Project Object	Body (JSON)	yes	Project object. Check the project data model definitions from the swagger documentation https://grano.emmi.fi/papi/v1/swagger/ui/index#!/Project/POST_project
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: Project Id (string).

cURL example creating a new project to a root hierarchy.

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/project' \
--header 'Content-Type: application/json' \
--header 'x-auth-token: [Session token]' \
--data-raw '{
    "Id": "0",
    "ParentId": "0",
    "Name": "My First EMMi Project",
    "ScheduleStart": 1642501733,
    "ScheduleEnd": 1642674533
}'
```

cURL example of changing the project name, moving it to archive named folder and changing status to finished.

Note that the project statuses can vary between EMMi instances

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/project' \
--header 'Content-Type: application/json' \
--header 'x-auth-token: [Session token]' \
--data-raw '{
    "Id": "123",
```

```
    "ParentId":"206",
    "StatusId": "8",
    "Name": "My First Project - Done"
}'
```

Get Project by Id

Request : GET /papi/v1/project/[Project id]

Function for getting the basic information of a project.

Parameter	Parameter Type	Mandatory	Description
Project Id	Path	yes	Unique project id.
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: JSON – Project object.

```
{
    "Id": "123",
    "ParentId": "0",
    "StatusId": "3",
    "Name": "My First Project",
    "ScheduleStart": 1642464000,
    "ScheduleEnd": 1646092799,
    "ScheduleFullDays": true,
    "DeadlineNotification": false,
    "Members": [
        {
            "ProjectRoleId": "1",
            "UserId": "3",
            "UserFullName": "John Average"
        }
    ],
    "PropertyValues": [
        {
            "FieldIdentifier": "81",
            "Value": true
        },
        {
            "FieldIdentifier": "82",
            "Value": {
```

```
                "x-default": "Lorem Brief."
            }
        }
    ],
    "ProofCount": 1,
    "AssetCount": 1,
    "MessageCount": 1,
    "Tasks": [
        {
            "Id": "204",
            "ParentId": "123",
            "Name": "My First Task",
            "Description": "Do something",
            "IsCompleted": false
        }
    ],
    "Milestones": [
        {
            "Id": "205",
            "ParentId": "123",
            "Name": "My First Milestone",
            "Description": "When it's done, it will be",
            "ScheduleEnd": 1645487999,
            "ScheduleFullDays": true,
            "DeadlineNotification": true
        }
    ]
}
```


cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/project/[Proje
ct id] \
--header 'x-auth-token: [Session token]'
```

3.2.2. Create & Get Folder

Create or Modify Folder

Request : GET /papi/v1/project/folder

Save a new folder or edit an existing one. If the folder Id = 0 or is not defined then the folder is saved as a new.

Parameter	Parameter Type	Mandatory	Description
Folder Object	Body	yes	Folder object. Check the folder data model definitions from the swagger documentation https://grano.emmi.fi/papi/v1/swagger/ui/index#!/Project/POST_project_folder
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: Folder Id (string).

cURL example creating a new folder to the root hierarchy.

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/project/folder' \
--header 'Content-Type: application/json' \
--header 'x-auth-token: [Session token]' \
--data-raw '{
    "Id": "0",
    "ParentId": "0",
    "Name": "My First EMMi Folder to Project Management"
}'
```

Get Project Folder Information

Request : GET /papi/v1/Folder/[Folder id]

Function for getting the basic information of a project folder. The folder object is identical to the project object excluding the proof, asset, task/milestone, conversation information.

Parameter	Parameter Type	Mandatory	Description
Folder Id	Path	yes	Unique folder id.

x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: JSON – Folder object.

```
{
    "Id": "75",
    "ParentId": "0",
    "Name": "Folder for Amazing Projects",
    "Members": [
        {
            "ProjectRoleId": "1",
            "UserId": "25",
            "UserFullName": "Jonh Average"
        },
        {
            "ProjectRoleId": "2",
            "UserId": "61",
            "UserFullName": "Mary Mediocre"
        }
    ],
    "PropertyValues": [
        {
            "FieldIdentifier": "43",
            "Value": {
                "x-default": "This is a description of the project."
            }
        }
    ]
}
```

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/Folder/[Folde
r id] \
--header 'x-auth-token: [Session token]'
```

3.2.3. Proof Information

Get proof information

Request : GET /papi/v1/proof/[Proof Id]

Get more detailed information about a proof.

Parameter	Parameter Type	Mandatory	Description
Proof Id	Path	yes	Unique proof Id to get the information.
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: JSON – Proof Object.

Proof Id JSON path: **$.Id**
Latest version file version Id JSON path: **$.CurrentVersionId**
All fileversion Id's JSON path: **$.Versions..Id**


```
{
    "Id": "2514",
    "Name": "EMMi proof",
    "ProjectId": "174",
    "ProofStatusId": "6",
    "Created": 1642082033,
    "Modified": 1642146967,
    "CurrentVersionId": "1463",
    "Versions": [
        {
            "Id": "1462",
            "Filename": "EMMi proof",
            "Extension": "png",
            "Created": 1642082033,
            "Size": 104229,
            "PreviewSupported": "Ready",
            "Info": {
                "Height": 564.0,
                "Width": 1424.0,
                "Unit": "px",
                "Resolution": 144.0,
                "Colorspace": "RGB"
            },
```

```
            "Activity": {
                "CommentCount": 0,
                "ApprovalCount": 0,
                "ConditionalApprovalCount": 0,
                "RejectCount": 0,
                "SharingGuestCount": 0,
                "IsSharingActive": false
            }
        },
        {
            "Id": "1463",
            "Filename": "EMMi proof v2",
            "Extension": "png",
            "Created": 1642082042,
            "Size": 34129,
            "PreviewSupported": "Ready",
            "Info": {
                "Height": 724.0,
                "Width": 507.0,
                "Unit": "px",
                "Colorspace": "RGB"
            },
            "Activity": {
                "CommentCount": 1,
                "ApprovalCount": 0,
                "ConditionalApprovalCount": 0,
                "RejectCount": 0,
                "SharingGuestCount": 0,
                "IsSharingActive": false
            }
        }
    ]
}
```


cURL example

```
curl --location --request GET 'https://[EMMi instance]/papi/v1/proof/[Proof I
d] \
--header 'x-auth-token: [Session token]'
```

3.2.4. Upload Proof

Request : POST /papi/v1/project/[Project Id]/proof/upload/[File name]

Upload a File as a New Proof to the Project. When the new proof is created a proof id is returned of a successful request.

Parameter	Parameter Type	Mandatory	Description
Project Id	Path	yes	Unique project id where to upload the file.
File name	Path	yes	Filename containing the extension.
File content	Body	yes	Request body containing the file as a binary file.
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: Proof Id (string).

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/project/[Proje
ct Id]/proof/upload/[File name]' \
--header 'Content-Type: application/octet-stream' \
--header 'x-auth-token: [Session token]' \
--data-binary '@[Path + File name]'
```

Delete proof

Deletes proof and its versions with the given ID.

Request : DELETE /papi/v1/proof/[Proof Id]

Parameter	Parameter Type	Mandatory	Description
Proof Id	Path	yes	Unique proof id
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: HTTP Status Code 204 – No Content

cURL example

```
curl --location --request DELETE 'https://[EMMi instance]/papi/v1/proof/[Proo
f Id]' \
--header 'x-auth-token: [Session token]'
```

3.2.5. Upload Proof Version

Request : POST /papi/v1/proof/[Proof Id]/upload/[File name]

Upload a new file as a new version to a proof. The given filename does not change the proof name. Proof name can be only modified by the Modify Proof command.

Parameter	Parameter Type	Mandatory	Description
Proof Id	Path	yes	Unique proof id of a proof to create the new version
File name	Path	yes	Filename containing the extension.
File content	Body	yes	Request body containing the file as a binary file.
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: Proof Id (string).

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/proof/[Proof Id]/upload/[File name]' \
--header 'Content-Type: application/octet-stream' \
--header 'x-auth-token: [Session Token]' \
--data-binary '@[Path + File name]'
```

Delete the latest proof version

Deletes the latest version of proof with the given proof Id and file version Id. Every proof version has a file version Id. The current file version Id can be fetched with the Get Proof command.

Request : DELETE /papi/v1/proof/[Proof Id]/[Version Id]

Parameter	Parameter Type	Mandatory	Description
Proof Id	Path	yes	Unique proof id
Version Id	Path	yes	File version id
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: HTTP Status Code 204 – No Content

cURL example

```
curl --location --request DELETE 'https://[EMMi instance]/papi/v1/proof/[Proo
f Id]/[Version Id]' \
--header 'x-auth-token: [Session token]'
```

3.2.6. Modify Proof

Request : POST /papi/v1/proof

Modify Proof Name, Description or Status. Other properties cannot be edited at this moment.
To get the available proof status id's, use the Get Proof Status command.

Parameter	Parameter Type	Mandatory	Description
Proof Object	Body	yes	Proof Id and the value(s) to change. Check the proof data model definitions from the swagger documentation https://grano.emmi.fi/papi/v1/swagger/ui/index#!/Project/POST_proof
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: Proof Id (string).

Examples

Change the proof name to "Amazing Product Catalogue"

```
{
    "Id": "[Proof Id]",
    "Name": "Amazing Product Catalogue"
}
```

Change Proof status to "Waiting for Comments"

```
{
    "Id": "[Proof Id]",
    "ProofStatusId": 4
}
```

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/proof' \
--header 'Content-Type: application/json' \
--header 'x-auth-token: [Session Token]' \
--data-raw '{
    "Id": [Proof Id],
    "ProofStatusId": 4
}'
```

3.2.7. Get list of proof statuses

Request : GET /papi/v1/proof/status

List of proofs statuses, with translations and properties. The proof id is used when changing the proof status via API. The proof statuses cannot be modified using the API.

> Note that the status Id's can be different between EMMi instances.

Parameter	Parameter Type	Mandatory	Description
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: JSON – A list of statuses.

```
[
    {
        "Id": "4",
        "Name": {
            "fi": "Odottaa kommentteja",
            "en": "Waiting for comments",
            "ru": "Waiting for comments",
            "sv": "Waiting for comments"
        },
        "IsSharingAllowed": true
    },
    {
        "Id": "5",
        "Name": {
            "fi": "Odottaa hyväksyntää",
            "en": "Waiting for approval",
            "ru": "Waiting for approval",
            "sv": "Waiting for approval"
        },
        "IsSharingAllowed": true
    }
]
```

cURL example

```
curl --location --request GET 'https://[EMMi instance]/papi/v1/proof/status' \
```

```
--header 'x-auth-token: [Session token]'
```

3.2.8. Proof Sharing

Share a proof

Proofs can be shared to external users outside the project for commenting. Sharing creates a personal link to the proof, linked to the guest user's email address. Guest users will not see any other project information and will not be able to log in to Mediabank.

There are two options to share the proof to quests:

- The share URL can be shared using the email invite sent by the system.
- The share URL can be fetched from the response and shared in desired way.

> Proof sharing can be combined with the Project Management Webhook to get information about the events in the proof. More about the webhook…

Request : /papi/v1/proof/share

Parameter	Parameter Type	Mandatory	Description
Share Object	Body	yes	Proof share object. Check the share data model definitions from the swagger documentation https://grano.emmi.fi/papi/v1/swagger/ui/index#!/Project/POST_proof_share
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: JSON – Share object.

```
{
    "ProofId": "321",
    "ProofStatusId": "4",
    "Guests": [
        {
            "Name": "Average Guest",
            "Email": "average.person@mail.com",
            "IsOpinionAllowed": true,
            "SendEmailInvitation": false,
            "IsActive": true,
            "ShareUrl": "https://[EMMi instance]/l/B897AYu3uWYh",
            "VersionIds": [
                "1234"
```

```
            ],
            "AccessCount": 0,
            "Invited": 1657884503,
            "Response": "Pending",
            "CommentCount": 0
        }
    ],
    "IsCancelled": false
}
```

cURL Examples

Sharing proof by sending the invite via email.

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/proof/share' \
--header 'x-auth-token: [Session token]' \
--header 'Content-Type: application/json' \
--data-raw '{
    "ProofId": "321",
    "ProofStatusId": "4",
    "Guests": [
        {
            "Name": "John Average",
            "Email": "john.average@mail.com",
            "IsOpinionAllowed": true,
            "SendEmailInvitation": true,
            "IsActive": true
        },
        {
            "Name": "Mary Mediocre",
            "Email": "mary.mediocre@mail.com",
            "IsOpinionAllowed": true,
            "SendEmailInvitation": true,
            "IsActive": true
        }
    ],
    "EmailMessageContent": "Hi,\n\nHere is a layout for you to review.\n\nCrea
ted by Proof Integration",
    "EmailLanguageCode": "en",
    "IsCancelled": false
}'
```

Email invite created by the request

The inviter name and email is from the API user who created the invite.

> Proof Integration (integration@mail.com) invited you to comment a proof in Grano project management.
>
> PROOF DETAILS:
> Proof name: New_amazing_catalog_128pages
>
> MESSAGE FROM Project Management
> Hi,
>
> Here is a layout for you to review.
>
> Created by Proof Integration
>
> You can accept or decline the invitation using this link:
> https://[EMMi instance]/l/B897AYu3uWYh
> This is your unique link to shared proof. Please keep in mind, that if you share this link forward, others can access the proof as you.

Sharing proof without sending an email invite.

```
 curl --location --request POST 'https://[EMMi instance]/papi/v1/proof/share'
\
--header 'x-auth-token: [Session token]' \
--header 'Content-Type: application/json' \
--data-raw '{
    "ProofId": "321",
    "ProofStatusId": "4",
    "Guests": [
        {
            "Name": "John Average",
            "Email": "john.average@mail.com",
            "IsOpinionAllowed": true,
            "SendEmailInvitation": false,
            "IsActive": true
        },
        {
            "Name": "Mary Mediocre",
            "Email": "mary.mediocre@mail.com",
            "IsOpinionAllowed": true,
            "SendEmailInvitation": false,
            "IsActive": true
        }
    ],
    "IsCancelled": false
}'
```

Cancel sharing for one user

```
 curl --location --request POST 'https://[EMMi instance]/papi/v1/proof/share'
\
--header 'x-auth-token: [Session token]' \
--header 'Content-Type: application/json' \
--data-raw '{
    "ProofId": "321",
    "Guests": [
        {
            "Email": "average.person@mail.com",
            "IsActive": false
        },
        {
            "Email": "mediocre.person@mail.com",
            "IsActive": true
        }
    ]
}'
```

Get share information

Request : GET papi/v1/proof/[Proof Id]/share

Function for getting the sharing Information of a shared proof.

Parameter	Parameter Type	Mandatory	Description
Proof Id	Path	yes	Unique proof id.
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: JSON – Share object.

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/proof/[Proof I
d]/share \
--header 'x-auth-token: [Session token]'
```

3.2.9. Get Conversations

Request : GET /papi/v1/project/[Project id]/message

Function for listing the all conversations inside of a project. Deleted conversations show in the list with the parameter ***"IsDeleted": true***.

Parameter	Parameter Type	Mandatory	Description
Project Id	Path	yes	Unique project id.
x-auth-token	Header	yes	The authentication token must be provided as an x-auth-token header for every API request. More information about authentication token.

Response: JSON – Project conversation object(s) as an array.


```
[
    {
        "Id": "16",
        "ProjectId": "321",
        "Subject": "Hello does this work?",
        "Body": "Hi,<br><br>Just testing this.<br><br>Br.<br>John Average",
        "Creator": {
            "UserId": "123",
            "UserFullName": "John Average"
        },
        "Created": 1642143617,
        "IsDeleted": false
    },
    {
        "Id": "17",
        "ParentId": "16",
        "ProjectId": "321",
        "Body": "Hi John!<br><br>Works Great.<br><br>Br.<br>Mary Mediocre",
        "Creator": {
            "UserId": "3",
            "UserFullName": "Mary Mediocre"
        },
        "Moderator": {
            "UserId": "3",
            "UserFullName": "Mary Mediocre"
        },
        "Created": 1642143694,
        "Modified": 1642143704,
```

```
        "IsDeleted": false
    }
]
```

cURL example

```
curl --location --request POST 'https://[EMMi instance]/papi/v1/project/[Proje
ct id]/message \
--header 'x-auth-token: [Session token]'
```