

API DOCUMENTATION GUIDE

Table of Contents

1	Introduction	- 2 -
2	Authentication	- 2 -
3	http Header for the Rest API.....	- 3 -
4	Create a user.....	- 4 -
4.1	Step1: Get the User Creation Form	- 4 -
4.2	Step2: Create a user	- 5 -
5	Get Users	- 6 -
6	Get a specific user	- 8 -
6.1	Get a profile summary, and the fields categories	- 8 -
6.2	Get the user profile per category.....	- 8 -
6.3	Get the whole user profile	- 9 -
7	Update a user.....	- 10 -
7.1	Edit a Dropdown:	- 10 -
7.2	Edit a Composite field.....	- 10 -
8	Data Types	- 11 -
8.1	Dropdown	- 11 -
8.2	Hierarchy	- 12 -
8.3	Date	- 13 -
8.4	File.....	- 13 -
8.5	Relation.....	- 16 -
8.6	Numeric.....	- 17 -
8.7	Composite	- 17 -
8.8	ListOf.....	- 19 -
8.8.1	Append a new value.....	- 20 -
8.8.2	Get the list of values.....	- 20 -
8.8.3	Edit an existing value from the list	- 20 -
9	Field Settings.....	- 21 -
10	Recommendations and notes	- 22 -

1 Introduction

Old documentation: <https://rest.monportailrh.com/swagger/>

New documentation: <https://rest.monportailrh.com/docs/>

2 Authentication

To get a Bearer token, you should use the API request below:

POST <https://sso.monportailrh.com/auth/realms/Internal-ldap/protocol/openid-connect/token>

Header: Content-Type = application/x-www-form-urlencoded

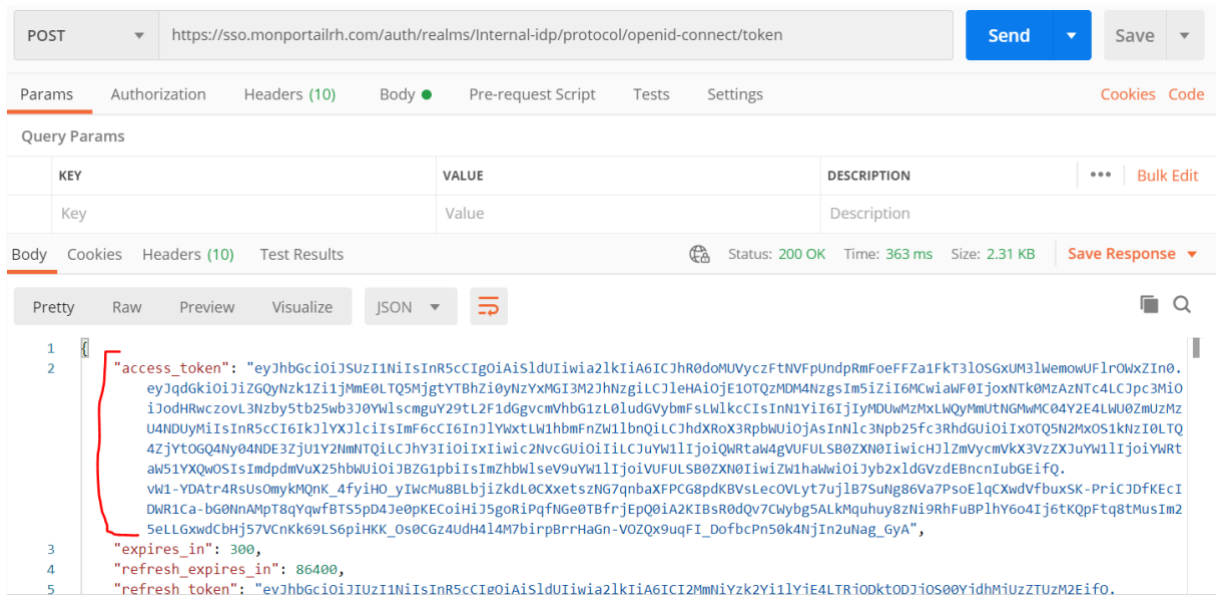
Body:

The screenshot shows a REST client interface with the following details:

- Method:** POST
- URL:** https://sso.monportailrh.com/auth/realms/Internal-ldap/protocol/openid-connect/token
- Buttons:** Send, Save
- Tabs:** Params, Authorization, Headers (10), Body (selected), Pre-request Script, Tests, Settings, Cookies, Code
- Body Type:** x-www-form-urlencoded (selected)
- Parameters Table:**

KEY	VALUE	DESCRIPTION
☒ grant_type	password	
☒ client_id	realm-management	
☒ username	[REDACTED]	
☒ password	[REDACTED]	
Key	Value	Description

API response:



Token expires within 5 minutes, during this time you can:

1. Login again
2. Or, Refresh the token by sending following request:
 - POST <https://sso.monportailrh.com/auth/realms/Internal-idp/protocol/openid-connect/token>
 - Header: "Content-Type" = application/x-www-form-urlencoded
 - Body:
 - grant_type: refresh_token
 - client_id: realm-management
 - refresh_token: {the expired token}

3 http Header for the Rest API

All the API requests to <https://rest.monportailrh.com>.

URL should contain following headers:

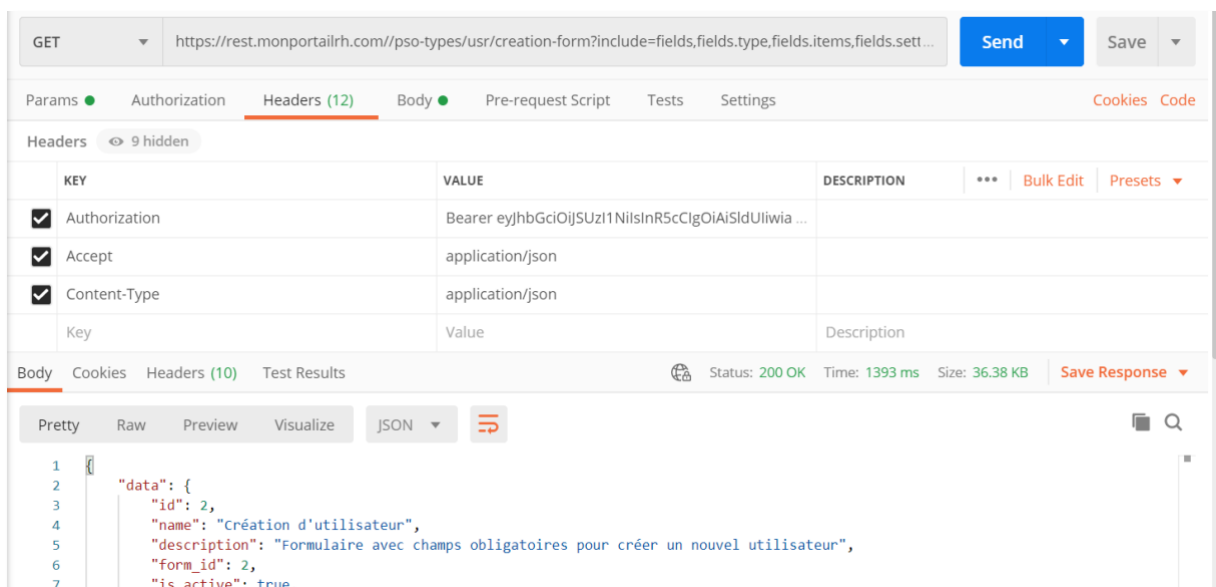
☒	Authorization	Bearer eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUiIiwia ...
☒	Accept	application/json
☒	Content-Type	application/json

4 Create a user

The user creation has to be done on two steps:

4.1 Step 1: Get the User Creation Form

GET https://rest.monportailrh.com//pso-types/usr/creation-form?include=fields,fields.type,fields.items,fields.settings,fields.options,fields.assignment_settings,fields.items.type,fields.items.option



4.2 Step 2: Create a user

POST <https://rest.monportailrh.com/pso-types/usr>

Body:



Body in Json format:

```
{
  "data": [
    {
      "field_id": 272,
      "value": "228"
    },
    {
      "field_id": 550,
      "value": "321"
    },
    {
      "field_id": 123,
      "value": "Riad"
    },
    {
      "field_id": 125,
      "value": "Lasfer"
    },
    {
      "field_id": 127,
      "value": "RiadLasferTestJuly9"
    },
    {
      "field_id": 28,
      "value": "RiadLasferTestJuly9@gmail.com"
    },
    {
      "field_id": 25,
      "value": "00000"
    },
    {
      "field_id": 27,
      "value": "1111"
    },
    {
      "field_id": 182,
      "value": "6"
    },
    {
      "field_id": 344,
      "value": "268"
    },
    {
      "field_id": 185,
      "value": "284"
    },
    {
      "field_id": 711,
      "value": "2020-07-10"
    },
    {
      "field_id": 186,
      "value": "285"
    },
    {
      "field_id": 199,
      "value": "74"
    },
    {
      "field_id": 540,
      "value": "2020-07-16"
    },
    {
      "field_id": 542,
      "value": "2020-07-16"
    },
    {
      "field_id": 544,
      "value": "111.1"
    },
    {
      "field_id": 546,
      "value": "2020-07-10"
    },
    {
      "field_id": 548,
      "value": "2020-07-10"
    },
    {
      "field_id": 716,
      "value": "2020-07-16"
    },
    {
      "field_id": 718,
      "value": "417"
    },
    {
      "field_id": 720,
      "value": "458"
    },
    {
      "field_id": 722,
      "value": "400"
    },
    {
      "field_id": 724,
      "value": "111"
    },
    {
      "field_id": 726,
      "value": "111"
    },
    {
      "field_id": 728,
      "value": "aaaa"
    },
    {
      "field_id": 730,
      "value": "aaaaa"
    },
    {
      "field_id": 732,
      "value": "2020-07-31"
    },
    {
      "field_id": 734,
      "value": "aaaa"
    },
    {
      "field_id": 708,
      "value": "383"
    }
  ]
}
```

5 Get Users

Get users per pagination, the request below is to get the first 25 active users

GET <https://rest.monportailrh.com/search?name=&pso-type=usr&page=1&active=1&per-page=25>

- pso-type=usr
- page=1
- active=1
- per-page=25

This API helps you to retrieve the list of users with the main fields :

- ID
- Email
- First name
- Last name
- ...

Once you retrieve the list of users, you can use the id to get the whole profile of a specific user, using the API below.

```

▼ data: [{...}]
  ▼ 0: Object { meta: {...}, pso_type: {...}, data: [...] }
    ▼ meta: Object { pagination: {...} }
      ▼ pagination: Object { total: 22, count: 22, per_page: 25, ... }
        total: 22
        count: 22
        per_page: 25
        current_page: 1
        total_pages: 1
        links: []
      ▶ pso_type: Object { id: 1, name: "User", alias: "usr", ... }
    ▼ data: [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, ...]
      ▶ 0: Object { id: 6, status: "default", role: [...], ... }
      ▶ 1: Object { id: 7, status: "default", role: [...], ... }
      ▶ 2: Object { id: 8, status: "default", role: [...], ... }
      ▶ 3: Object { id: 9, status: "default", role: [...], ... }
      ▶ 4: Object { id: 10, status: "default", role: [...], ... }
      ▶ 5: Object { id: 11, status: "default", role: [...], ... }
      ▶ 6: Object { id: 12, status: "default", role: [...], ... }
      ▶ 7: Object { id: 13, status: "default", role: [...], ... }
      ▶ 8: Object { id: 20, status: "default", role: [...], ... }
      ▶ 9: Object { id: 23, status: "default", role: [...], ... }
      ▶ 10: Object { id: 25, status: "default", role: [...], ... }
      ▶ 11: Object { id: 27, status: "default", role: [...], ... }
      ▶ 12: Object { id: 38, status: "default", role: [...], ... }
      ▶ 13: Object { id: 39, status: "default", role: [...], ... }
      ▶ 14: Object { id: 41, status: "default", role: [...], ... }
      ▶ 15: Object { id: 47, status: "default", role: [...], ... }
      ▶ 16: Object { id: 51, status: "default", role: [...], ... }
      ▶ 17: Object { id: 56, status: "default", role: [...], ... }
      ▶ 18: Object { id: 57, status: "default", role: [...], ... }
      ▶ 19: Object { id: 58, status: "default", role: [...], ... }
      ▶ 20: Object { id: 59, status: "default", role: [...], ... }
      ▶ 21: Object { id: 60, status: "default", role: [...], ... }
  
```

6 Get a specific user

You have to use the ID, to get the profile of a specific user.

6.1 Get a profile summary and the fields categories

GET

https://rest.monportailrh.com/psos/7?include=pso.pso_type.profile_form_instance_id,pso.categories,pso.fields

```

▼ data: Object { summary: {...}, pso: {...} }
  ▼ summary: Object { first_name: {...}, last_name: {...}, image: {...}, ... }
    ▶ first_name:
    ▶ last_name:
    ▶ image:
    ▶ is_active:
    ▶ email:
    ▶ phone_number: Object { id: 27, alias: "professional_mobile_phone", value: null }
  ▼ pso: Object { id: 7, pso_type: {...}, categories: [...], ... }
    id: 7
    ▶ pso_type: Object { id: 1, name: "User", alias: "usr", ... }
    ▼ categories: [ {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, ... ]
      ▶ 0: Object { id: 6, name: "Identity", alias: "identity" }
      ▶ 1: Object { id: 2, name: "Additional Identity information", alias: "additional_info" }
      ▶ 2: Object { id: 11, name: "Compensation", alias: "compensation" }
      ▶ 3: Object { id: 4, name: "Employment contract", alias: "employment_contract" }
      ▶ 4: Object { id: 5, name: "Family Identity", alias: "family_identity" }
      ▶ 5: Object { id: 7, name: "Leaves & Absences, Timesheets and Expenses", alias: "leaves_timesheet_expenses" }
      ▶ 6: Object { id: 14, name: "Performance", alias: "performance" }
      ▶ 7: Object { id: 9, name: "Personal Contact", alias: "personal_contact" }
      ▶ 8: Object { id: 3, name: "Professional Contact", alias: "contact" }
      ▶ 9: Object { id: 10, name: "Relation", alias: "relation" }
      ▶ 10: Object { id: 16, name: "Resume", alias: "cv" }
      ▶ 11: Object { id: 12, name: "Settings", alias: "settings" }
      ▶ 12: Object { id: 17, name: "Training", alias: "training" }
    ▼ fields: [ {...}, {...}, {...}, {...} ]
      ▶ 0: Object { id: 423, name: "Profile Picture", is_active: true, ... }
      ▶ 1: Object { id: 272, name: "Active", is_active: true, ... }
      ▶ 2: Object { id: 123, name: "First Name", is_active: true, ... }
      ▶ 3: Object { id: 125, name: "Last Name", is_active: true, ... }

```

6.2 Get the user profile per category

GET

https://rest.monportailrh.com/psos/7/fields?category=identity&active=true&include=type,items,options,settings,assignment_settings

```
▼ data: [ {}, {}, {}, {}, {}, {}, {}, {}, {}, {} ]
  ▶ 0: Object { id: 851, name: "Group", is_active: true, ... }
  ▶ 1: Object { id: 276, name: "User ID", is_active: true, ... }
  ▶ 2: Object { id: 423, name: "Profile Picture", is_active: true, ... }
  ▶ 3: Object { id: 272, name: "Active", is_active: true, ... }
  ▶ 4: Object { id: 550, name: "Civility", is_active: true, ... }
  ▶ 5: Object { id: 123, name: "First Name", is_active: true, ... }
  ▶ 6: Object { id: 125, name: "Last Name", is_active: true, ... }
  ▶ 7: Object { id: 127, name: "Username", is_active: true, ... }
  ▶ 8: Object { id: 549, name: "Access card number", is_active: true, ... }
  ▶ 9: Object { id: 439, name: "Disabled Worker", is_active: true, ... }
  ▶ 10: Object { id: 428, name: "Driver's licence", is_active: true, ... }
  ▶ 11: Object { id: 552, name: "Social security number", is_active: true, ... }
  ▶ 12: Object { id: 553, name: "Birth", is_active: true, ... }
  ▶ 13: Object { id: 310, name: "Nationality", is_active: true, ... }
  ▶ 14: Object { id: 850, name: "Matricule", is_active: true, ... }
  ▶ 15: Object { id: 920, name: "file", is_active: true, ... }
```

6.3 Get the whole user profile

GET https://rest.monportailrh.com/psos/7/fields?active=true&include=type,items,options,settings,assignment_settings

```
▼ data: [ {}, {}, {}, {}, {}, {}, {}, {}, {}, {} ]
  ▶ 0: Object { id: 851, name: "Group", is_active: true, ... }
  ▶ 1: Object { id: 276, name: "User ID", is_active: true, ... }
  ▶ 2: Object { id: 423, name: "Profile Picture", is_active: true, ... }
  ▶ 3: Object { id: 272, name: "Active", is_active: true, ... }
  ▶ 4: Object { id: 550, name: "Civility", is_active: true, ... }
  ▶ 5: Object { id: 123, name: "First Name", is_active: true, ... }
  ▶ 6: Object { id: 125, name: "Last Name", is_active: true, ... }
  ▶ 7: Object { id: 127, name: "Username", is_active: true, ... }
  ▶ 8: Object { id: 549, name: "Access card number", is_active: true, ... }
  ▶ 9: Object { id: 439, name: "Disabled Worker", is_active: true, ... }
  ▶ 10: Object { id: 428, name: "Driver's licence", is_active: true, ... }
  ▶ 11: Object { id: 552, name: "Social security number", is_active: true, ... }
  ▶ 12: Object { id: 553, name: "Birth", is_active: true, ... }
  ▶ 13: Object { id: 310, name: "Nationality", is_active: true, ... }
  ▶ 14: Object { id: 850, name: "Matricule", is_active: true, ... }
  ▶ 15: Object { id: 920, name: "file", is_active: true, ... }
  ▶ 16: Object { id: 574, name: "Personal car", is_active: true, ... }
  ▶ 17: Object { id: 929, name: "aaaa", is_active: true, ... }
  ▶ 18: Object { id: 930, name: "url", is_active: true, ... }
  ▶ 19: Object { id: 697, name: "Salary structure", is_active: true, ... }
  ▶ 20: Object { id: 626, name: "Health Insurance", is_active: true, ... }
  ▶ 21: Object { id: 688, name: "Travel", is_active: true, ... }
  ▶ 22: Object { id: 612, name: "Bank Account", is_active: true, ... }
  ▶ 23: Object { id: 28, name: "Professional email address", is_active: true, ... }
  ▶ 24: Object { id: 25, name: "Professional Phone", is_active: true, ... }
  ▶ 25: Object { id: 27, name: "Professional Mobile Phone", is_active: true, ... }
  ▶ 26: Object { id: 450, name: "Business Address", is_active: true, ... }
```

7 Update a user

You have to use the id, to edit the profile of a specific user

7.1 Edit a Dropdown:

POST <https://rest.monportailrh.com/psos/7/form-instances/1>

Body in Json format: {"data":[{"field_id":550,"value":["322"]}]}

7.2 Edit a Composite field

POST <https://rest.monportailrh.com/psos/7/form-instances/1>



Body in Json format:

```

{"data":[{"field_id":441,"value":["1"]}, {"field_id":443,"value":["293"]}, {"field_id":445,"value":["2020-07-10"]}, {"field_id":447,"value":["2020-07-17"]}, {"field_id":786,"value":["FLATCHR LOGO.png"]}, {"field_id":789,"value":["https://rest.monportailrh.com/storage/demo_uat09/files/4a/59/Szg1dMXEaOTYJi9ejled6L1taVzCikdXVXhDDd5w5f073d79e4470.png"]}]}

```

8 Data Types

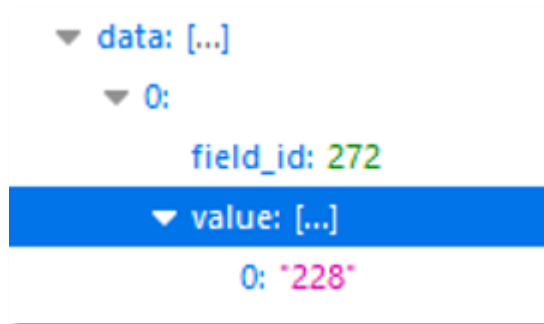
8.1 Dropdown

```

▼ 0: Object { id: 272, name: "Active", is_active: true, ... }
  id: 272
  name: "Active"
  is_active: true
  removable: false
  has_unique_data: false
  list: false
  read_only: false
  alias: "usr_active"
  position: 1
  is_locked: true
  required: true
  values_count: 0
  has_value: true
  ▶ access_permissions: Object { hide: false, view: true, edit: true, ... }
    values: []
    value_details: []
  ▶ type: Object { id: 4, alias: "list", name: "Single choice dropdown", ... }
    items: []
▼ options: [{...}, {...}]
  ▼ 0:
    id: 228
    position: 1
    value: "228"
    name: "Yes"
    is_active: true
  ▼ 1:
    id: 229
    position: 2
    value: "229"
    name: "No"
    is_active: true
  settings: []
  ▶ assignment_settings: [{...}, {...}, {...}]

```

To use the first option « Yes », you have to use the id of this option in your API request « 228 »



8.2 Hierarchy

The Hierarchy type is similar to the dropdown type, and each parent option can have more options as children.



To use an option as a value, you have to use the id of this option in your API request

8.3 Date

Date format: yyyy-MM-dd

8.4 File

The File type has the same structure as a Composite field.

```

▼ 1: Object { id: 920, name: "file", is_active: true, ... }
  id: 920
  name: "file"
  is_active: true
  removable: true
  has_unique_data: false
  list: false
  read_only: false
  alias: "usr_file"
  position: 59
  is_locked: false
  required: false
  values_count: 0
  has_value: false
  ▶ access_permissions: Object { hide: false, view: true, edit: true, ... }
    values: []
    value_details: []
  ▶ type: Object { id: 23, alias: "named_file", name: "File", ... }
  ▶ category: Object { id: 6, alias: "identity", position: 0, ... }
  ▶ pso_type: Object { id: 1, name: "User", alias: "usr", ... }
  ▼ items: [ {...}, {...} ]
    ▶ 0: Object { id: 922, name: "file name", is_active: true, ... }
    ▶ 1: Object { id: 924, name: "file link", is_active: true, ... }
    options: []
    settings: []
  ▶ assignment_settings: [ {...} ]

```

It contains 2 items:

1. File name

```
▼ items: [{...}, {...}]
  ▼ 0: Object { id: 922, name: "file name", is_active: true, ... }
    id: 922
    name: "file name"
    is_active: true
    removable: true
    has_unique_data: false
    list: false
    read_only: false
    alias: "usr_file_name1"
    position: 1
    values_count: 0
    has_value: true
    access_permissions: null
    values: []
    value_details: []
    ▶ type: Object { id: 1, alias: "string", name: "Short Text", ... }
    ▶ category: Object { id: 6, alias: "identity", position: 0, ... }
    ▶ pso_type: Object { id: 1, name: "User", alias: "usr", ... }
    items: []
    options: []
    settings: []
```

2. File Link

```
▼ 1: Object { id: 924, name: "file link", is_active: true, ... }
  id: 924
  name: "file link"
  is_active: true
  removable: true
  has_unique_data: false
  list: false
  read_only: false
  alias: "usr_file_link1"
  position: 2
  values_count: 0
  has_value: true
  access_permissions: null
  values: []
  value_details: []
  ▶ type: Object { id: 13, alias: "url", name: "Url", ... }
  ▶ category: Object { id: 6, alias: "identity", position: 0, ... }
  ▶ pso_type: Object { id: 1, name: "User", alias: "usr", ... }
  items: []
  options: []
  settings: []
```

To attach a File to a user profile, you need to follow these 2 steps:

- Step 1: upload the File

POST <https://rest.monportailrh.com/files/forms>

[illegible]

API response:

```
data: Object { url: "https://rest.monportailrh.com/storage/demo_uat09/files/4a/59/Szg1dMXEaOTYji9ejled6L1taVzCikdXVXhDDd5w5f073d79e4470.png", name: "FLATCHR LOGO.png" }
url: "https://rest.monportailrh.com/storage/demo_uat09/files/4a/59/Szg1dMXEaOTYji9ejled6L1taVzCikdXVXhDDd5w5f073d79e4470.png"
name: "FLATCHR LOGO.png"
```

The API will return you the File name and the File Link that should be used as a field value.

- Step 2: Edit the File Field

POST <https://rest.monportailrh.com/psos/7/form-instances/1>

```

▼ data: [...]
  ▼ 0:
    field_id: 922
    ▼ value: [...]
      0: "FLATCHR LOGO.png"
  ▼ 1:
    field_id: 924
    ▼ value: [...]
      0: "https://rest.monportailrh.com/storage/demo_uat09/files/08/4b/SbjSAABgSo1G8RdU9RSCIJxtgo93Hka1Y6UNNAfQ5f0740925a30e.png"

```

Body in Json format: `{"data":[{"field_id":922,"value":["FLATCHR LOGO.png"]}, {"field_id":924,"value":["https://rest.monportailrh.com/storage/demo_uat09/files/08/4b/SbjSAABgSo1G8RdU9RSCIJxtgo93Hka1Y6UNNAfQ5f0740925a30e.png"]}]}`

8.5 Relation

For the Relation type, like « Manager » can be used to link a user to his manager.

```

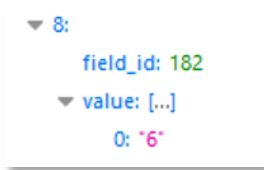
▼ 11: Object { id: 182, name: "Manager", is_active: true, ... }
  id: 182
  name: "Manager"
  is_active: true
  removable: false
  has_unique_data: false
  list: false
  read_only: false
  alias: "manager"
  position: 9
  is_locked: true
  required: false
  values_count: 0
  has_value: true
  ▶ access_permissions: Object { hide: false, view: true, edit: true, ... }
    values: []
    value_details: []
  ▶ type: Object { id: 12, alias: "user_relation", name: "PSO Relation", ... }
    items: []
  ▶ settings: [ {...}, {...} ]
    options: []
  ▶ assignment_settings: [ {...}, {...} ]

```

To link a user A to his manager (user B):

1. You can do it during the user creation of the user A, only if the user B already exists, else you need to create the user B first;

- If the user B already exists, you can use his user_ID as a value of this field as seen below (id=6):



8.6 Numeric

It can be an integer, or a Double with 14 digits for the precision.

8.7 Composite

A Composite field is a set of subfields, called items
Below is how it looks on the web application:

Hiring *	Prior seniority (in months)	
	End of trial - notice period	MM/DD/YYYY
	Trial period end date	MM/DD/YYYY
	Contract effective date	MM/DD/YYYY
	Contract type	Select Option
	If temporary, reason	Select Option
	Working hours	Select Option
	Annual days	

And here is the API response:

```

▼ 8: Object { id: 538, name: "Hiring", is_active: true, ... }
  id: 538
  name: "Hiring"
  is_active: true
  removable: false
  has_unique_data: false
  list: false
  read_only: false
  alias: "usr_hiring"
  position: 15
  is_locked: false
  required: false
  values_count: 0
  has_value: false
  ▶ access_permissions: Object { hide: false, view: true, edit: true, ... }
    values: []
    value_details: []
  ▶ type: Object { id: 11, alias: "composite", name: "Composite", ... }
  ▼ items: [{...}, {...}, {...}, {...}, {...}]
    ▶ 0: Object { id: 540, name: "Hiring date", is_active: true, ... }
    ▶ 1: Object { id: 542, name: "Last hiring date", is_active: true, ... }
    ▶ 2: Object { id: 544, name: "Prior seniority (in months)", is_active: true, ... }
    ▶ 3: Object { id: 546, name: "End of trial - notice period", is_active: true, ... }
    ▶ 4: Object { id: 548, name: "Trial period end date", is_active: true, ... }
    options: []
    settings: []
  ▶ assignment_settings: [{...}]

```

8.8 ListOf

A ListOf field can be any field type (text, numeric, dropdown, etc.), we can append new values to the list, we can also edit/delete the existing values.

```

▼ data: [{...}]
  ▼ 0: Object { id: 934, name: "Test Hierarchy", is_active: true, ... }
    id: 934
    name: "Test Hierarchy"
    is_active: true
    removable: true
    has_unique_data: false
    list: true
    read_only: false
    alias: "usr_test_hierarchy"
    position: 64
    is_locked: false
    required: false
    values_count: 0
    has_value: false
    ▶ access_permissions: Object { hide: false, view: true, edit: true, ... }
      values: []
      value_details: []
    ▶ type: Object { id: 6, alias: "hierarchy", name: "Hierarchy", ... }
    ▶ category: Object { id: 4, alias: "employment_contract", position: 4, ... }
    ▶ pso_type: Object { id: 1, name: "User", alias: "usr", ... }
      items: []
    ▶ options: [{...}]
      settings: []
  ▼ assignment_settings: [{...}]
    ▶ 0:

```

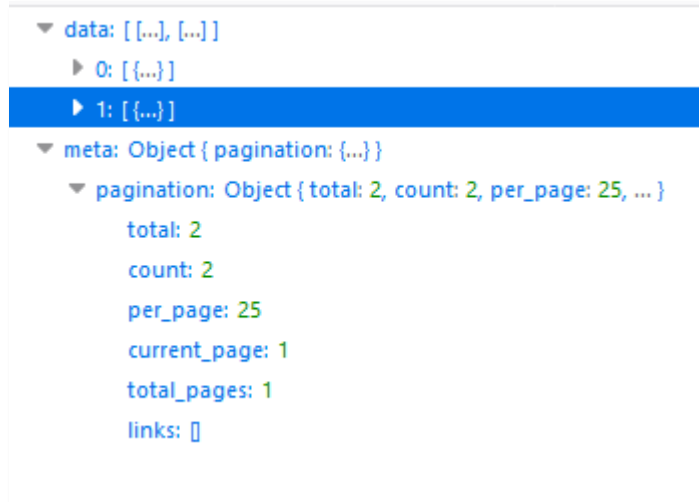
8.8.1 Append a new value

POST <https://rest.monportailrh.com/psos/7/form-instances/1/fields/934/values>

Body in Json format: {"data":[{"field_id":934,"value":["2841"]}]}

8.8.2 Get the list of values

GET <https://rest.monportailrh.com/psos/7/form-instances/1/fields/934/values?page=1>



8.8.3 Edit an existing value from the list

PATCH <https://rest.monportailrh.com/psos/7/form-instances/1/fields/934/values>

Body in Json format: {"data":[{"value_id":144154,"value":["2843"]}]}

9 Field Settings

- `id`: the ID is used as an identifier to edit the field through API
- `is_active`: it should be « true » to be able to edit it
- `has_unique_data`: if « true », the data should be unique across all users for this specific field
- `list`: if « true », this field can be used as a list of values, we can append a new value or edit/delete the existing values
- `required`: if « true », the field value is required in the API request
- `values_count`: counter of values for the `listOf` field

```
▼ 8: Object { id: 538, name: "Hiring", is_active: true, ... }
  id: 538
  name: "Hiring"
  is_active: true
  removable: false
  has_unique_data: false
  list: false
  read_only: false
  alias: "usr_hiring"
  position: 15
  is_locked: false
  required: false
  values_count: 0
  has_value: false
```

10 Recommendations and notes

- The API requests Body and API responses are dynamic, they depend on the customer configurations (Fields, Forms and Roles configurations);
- It's highly recommended to use an admin account, to have access to all Data, Forms and Fields;
- The options of dropdown/hierarchy fields are translatable, for the field « Active »
 - If the user language of the user who is used by the API is EN: the API will return Yes/No
 - If the user language is FR: the API will return Oui/Non
 So, you need to always keep the same user language for the user used by the API.
- If a Composite field is marked as required, all items should be included in request;
- « Username » and « Professional email address » should be unique;
- To retrieve all users and all listOf fields values, it's highly recommended to use the default export named « Rapport_Utilisateurs_customerName.zip » which is available on your sFTP following this path /customerName /incoming/reports.