Admin system REST API

This documents the possibility to interact with the admin system using REST calls (GET, POST, PUT, PATCH, DELETE) allowing automation of tasks like creating projects and users.

All examples are produced using the curl utility.

General

Authentification

The system uses BASIC AUTH authentification. Logging in simply need a admin system user and you can log in with the same username password combination that you uses for normal WEB access.

Data formats and URL's

All resources can be accessed using HTML, XML or JSON formats.

E.g. the link http://[url]/projects gives the normal HTML project listing while http://[url]/projects.xml gives the same list in XML format and http://[url]/projects.json gives the list with project and project users in JSON format.

Some resources (notable users) has dots (.) in their resource names so the .json can not be used. Be sure to send "Accept: application/json" in the header instead so

```
curl -H "Accept: application/json" -s -u testadmin:testpw http://localhost:3000/users/hakonhc
```

will give the same as

```
curl -s -u testadmin:testpw http://localhost:3000/users/hakonhc.json
```

Resources:

Resouce	Description
/projects	Lists and manipulation of projects
/owners	Listing of project owners
/users	Listing and manipulation of users on the server
/database	Listing and manipulation of databases
/project_users	Listing and manipulation of users in projects

Most resources responds to both GET, PATCH and DELETE

In addition we have some special endpoint for:

Resouce	Description
/node/logins	Get login statistics for projects for the last 5 years, grouped by type of client used(Revit, dRofus etc.)
/node/logins?from_date=2018-01-01&to_date=2019-12-31	Get login statistics for projects for the given time period, grouped by type of client used (Revit, dRofus etc.)
/node/unique_users	Gets number of unique users for each project for the last 5 years
/node/unique_users?from_date=2018-01-01&to_date=2019-12-31	Gets number of unique users for each project for the given period.
/project_data	Get statstics values on projects
/password/request_reset	A post request with a valid username will trigger a password reset email
/password/reset	Use this with a token to change password with the API

GET Resources: Listings

Resource	Description	Parameters
/projects	List all projects	*?query=xx* will list all projects contaning xx in the name ?show_all=1 to also include inactive projects
/projects/1	List project with id 1	
/owners	List all owners	
/owners/1	List owner with id 1	
/users	List of all users	?q=xx to search
/users/username	List user with username	
/database	List databases	
/project_users/username,projectid	Show project user	

Example

```
$ curl -s -u testadmin:testpw http://localhost:3000/projects.json?query=template|json_reformat
[
    {
        "project": {
            "active": true,
            "contact": null,
            "created_at": "2016-11-01T09:39:14Z",
            "created_by": null,
            "database_id": "akl-test",
            "description": null,
            "gross_area": null,
            "id": 399,
            "name": "dRofus dev template",
            "no": "01",
            "owner_id": 5,
            "status": null,
            "updated": null,
            "updated_by": null
        }
    },
.....
```

POST Resources: Creating objects

Example

This creates an owner with the name "Test" and from the return we can see that it has been assigned with ID 11

```
$ curl -H "Accept: application/json" -H "Content-type: application/json" -X POST -u testadmin:testpw -d
'{"owner":{"name":"Test"}}' http://localhost:3000/owners


{"owner":{"address":null,"billing_address":null,"contact":null,"id":11,"image":null,"name":"Test","network":
null,"note":null,"tech_contact":null}}
```

Creating a new project (POST /projects)

Creating a new project requires some special parameters. This would the minimal data to provide when creating a new project

Parameter	Descriptione
new_db	1 to create a new database or 0 to add project to an existing database
new_db_template	If creating a new database, provide the database name that would be used as a template
new_db_name	If creating a new database, provide the name of the new database
existing_db_name	If NOT creating a new database (*new_db* set to *0*) provide the name of the existing database to add the new project to
name	Name of the project
constructor	Name of the constructor/firm of the new project
description	Description of the project
owner_id	ID of the owner
project_type_id	Type of project. Use one of the following values id name 1 Active Project 2 Deprecated Project 3 Demo 4 Copy / Backup 5 Template 6 Sandbox / Test 7 Training 8 Trial

All the parameters are mandatory

Example

```
{
  "project": {
    "new_db": "1",
    "new_db_template": "dev-template",
    "new_db_name": "rest_test",
        "project_type_id": 1,
    "name": "REST TEST",
    "owner_id": 5,
    "description": "TEST CREATE FROM REST",
    "constructor" : "dRofus AS"
  }
}
```

Creating a new project user (POST /project_users)

```
$ curl -H "Accept: application/json" -H "Content-type: application/json; charset=UTF-8" -X POST -u testadmin:
testpw -d @data.json http://localhost:3000/project_users

{"project_user":{"created_at":"2016-11-28T12:22:35Z","project_id":408,"role":null,"superuser":null,"
user_role_id":null,"username":"hakonhc"}}
```

Wherer data.json contains

```
{
  "project_user": { "project_id": 408,"room_rights":1 },
  "user": {"username": "hakonhc", "first_name":"Håkon","last_name":"Clausen","email":"hhc@drofus.com"},
  "mail_type": "6"
}
```

Parameter	Descriptione
project_user	project_id: ID of the project to add to room_rights: room permission level equipment_rights tender_rights consignation_rights system_rights modelstore_rights superuser. Project administrator addon_admin: BIM admin no_web_admin_access: if user is superuser, this denies the access to the web admin
user	username: Username of the user to add first_name: First name of the user to add last_name: Last name of the user to add email: Email of the user to add *If the user exists, make sure that the information given is equal to the information registred on the server*
mail_type	ID of the email to send to the user (from /emails)

GET Resources: Object manipulation and more

Databases

Resource	
/database/[dbname]/disableall	Disables all project users in database with name dbname
/database/[dbname]/enableall	Enables all project users in database with name dbname
/database/[dbname]/kickall	Logs out all users in database with name dbname
/database/[dbname]/get_backup_now	Downloads a backup of the database

Users

Resource	
/users/[username]/disable	Disables a users so he can not log into any project
/users/[username]/enable	Enables a user
/users/[username]/kick	Logs out all users in database xxx

PUT/PATCH Resources: Updates

Resource	
/projects/[id]	Update project
/project_users/username,projectid	Update project user

Example 1: Update project

```
$ curl -H "Accept: application/json" -H "Content-type: application/json; charset=UTF-8" -X PATCH -u testadmin:
testpw -d @data.json http://localhost:3000/projects/1
```

Wherer data.json contains

```
{
  "project": {
    "name": "REST TEST",
    "description": "TEST UPDATE FROM REST",
        "active":true
  }
}
```

Example 2: Update project user

```
$ curl -H "Accept: application/json" -H "Content-type: application/json; charset=UTF-8" -X PATCH -u testadmin:
testpw -d @data.json http://localhost:3000/project_user/testuser,1
```

Wherer data.json contains

```
{
  "project_user":{
        "username":"test",
        "project_id":647,
        "room_rights":1,
        "equipment_rights":3,
        "tender_rights":4,
        "room_surface_treatment_rights":4
  }
}
```

DELETE Resources

You also use the DELETE operation to delete most object.

This example will delete a user in a project

```
curl -H "Accept: application/json" -s -u http://localhost:3000/project_users/[username],[project_id] -X DELETE
```

Resetting Password

Posting a json request to "/password/request_reset" with this body and a correct username will start the reset password process. The user will get a password reset request on email with a token:

```
// Post to {server}/password/request_reset
{
        "username": "user"
}
```

To actually change the password use the following json with the token from the email:

```
// Post to {server}/password/reset
{
        "token": "5f2pi2zW7wO3nodwWznmDQ",
        "password": "pass",
        "password_confirm": "pass"
}
```

Project statistics

Resource	Comment
/project_data	Gives project statistics over time
/project_data/latest	Gives you the latest project statistics for each project

Filters

from_date=[date]	Only for /project_data, gives you only data where time is greater than the given date
field=[field]	Only for specific field
owner=[owner_id]	Only for specific owner id

Example output

```
{
"field": "sum_programmed_area",
"project_id": 1262,
"time": "2019-12-16T10:56:46.848+01:00",
"value": "1233.0"
},
{
"field": "sum_designed_area",
"project_id": 1262,
"time": "2019-12-16T10:56:46.848+01:00",
"value": "1176.0"
},
```