Accessing AIQ Repository Configuration via REST APIs

This is a list of useful repository related REST APIs,

getRepositoryNamesByUser

Endpoint

rest/preferences/getRepositoryNamesByUser

Description

Returns an array with all repository configuration names from current logged in user.

Request parameters | payload

N/A

Response

`[ "repoName1", "repoName2", "repoName3" ]`
or empty array if no repo configs exists.

`[ ]`

getRepositoryByUser

Endpoint

`rest/preferences/getRepositoryByUser`

Description

Returns JSON object with details of the a repo given its name or empty JSON if no configuration with provided name is found.

Request parameters | payload

`?name=repoName1`

Response

Success


```
{     "proxyPort": 0,     "commitMessage": "Appvance Commit",     "auth":
"Password",     "proxyServer": "",     "proxyUsername": "",     "name":
"repoName1",     "proxyPassword": "",     "proxyType": "defatult",     "type":
"GIT",     "branch": "main",     "url": "https://gitlab.com/appvnc/repomemo/",     "username":
"appvnc"}
```


Error (repoName1 was not found)


```
{ }
```


getRepositoryStatus

Endpoint

`rest/preferences/getRepositoryStatus`

Description

Returns JSON object with status of the a repo given its name.

Request parameters | payload

`?name=repoName1`

Response

Success:

```
{     "timeStamp": "2023/01/30 19:59:52",     "stack": "",     "status": "WORKING"}
```

Error:

```
{     "timeStamp": "0",     "stack": "Repository not found",     "status": "ERROR"}
```

pullRepository

Endpoint

`rest/preferences/pullRepository`

Description

If repo is found, it triggers a git pull action on the repo and then executes a getRepositoryStatus returning the same JSON as that call.

Request parameters | payload

`?name=repoName1`

Response

Success:

```
{     "timeStamp": "2023/01/30 19:59:52",     "stack": "",     "status": "WORKING"}
```

On error:

{

```
{     "timeStamp": "2023/01/30 19:59:52",     "stack": "Repository not found",     "status": "ERROR"}
```

getBranches

Endpoint

`rest/preferences/getBranches`

Description

Returns the names of all the existing branches in the remote repository

Request parameters | payload

`?name=repoName1`

Response

Success:

```
{     "timeStamp": "0",     "stack": "",     "branches":
[     "main",     "performanceFolder"     ],     "status": "OK"}
```

On Error:

```
{     "timeStamp": "0",     "stack": "Repository not found",     "status": "ERROR"}
```

saveRepository

Endpoint

`rest/preferences/saveRepository`

Description

Saves new or updates repository configuration

Possible error messages:
"Repository with given name exists"
"Incorrect credentials."

Request parameters | payload

{"type":"GIT","name":"repoName1","url":"https://gitlab.net/ac/repo.git","username":"user@example.com","ssh":"","auth":"Password","password":"Wkdz!vxh14x9","branch":"","commitMessage":"","proxyServer":"","proxyPort":0,"proxyUsername":"","proxyPassword":"","proxyType":"defatult"}

Response

Success:

{ "message": "Repository successfully saved.", "status": "success"}

On Error:

{ "message": "Error: check your repository configuration.", "status": "error"}

deleteRepository

Endpoint

`rest/preferences/deleteRepository`

Description

Deletes a repository configuration named repoName1 for current user.

Request parameters | payload

`?repositoryName=repoName1`

Response

Success:

```
done true
```

On Error:

```
done false
```

Last updated: October 16, 2026

Revision #6
Created 2026-07-31 15:38:59 UTC by Admin
Updated 2026-10-16 18:59:49 UTC by mmora