Bitbucket rest api 2.0 documentation link:

https://developer.atlassian.com/bitbucket/api/2/reference/resource/

**Create project rest api**

**Endpoints url**:

[https://api.bitbucket.org/2.0/workspaces/{workspace}/projects](https://api.bitbucket.org/2.0/workspaces/%7bworkspace%7d/projects)

**Http method**: POST

**Path parameters**:

workspace: This can either be the workspace ID (slug) or the workspace UUID surrounded by curly-braces, for example: {workspace UUID}.

**Request header**:

Authentication: Basic Authentication

Content type: application/json

**Curl command:**

$ curl -H "Content-Type: application/json" \

-X POST \

-d "$body" \

https://api.bitbucket.org/2.0/teams/teams-in-space/projects/ | jq .

{

// Serialized project document

}

**Request body:**

{

"name":" ",

"key":" ",

"description":" ",

"links":{

"avtar":{

"href”:”<http://i.imgur.com/72tRx4w.gif>”

}

},

"is\_private":false

}

**Response Data:**

{

“name”:” ”,

“description”:””,

“is\_private”:false,

“created\_on”:””,

“updated\_on”:””

}

**Get All projects**

**Endpoints url**:

[https://api.bitbucket.org/2.0/workspaces/{workspace}/projects](https://api.bitbucket.org/2.0/workspaces/%7bworkspace%7d/projects)

**Http method**: GET

**Path parameters**:

workspace: This can either be the workspace ID (slug) or the workspace UUID surrounded by curly-braces, for example: {workspace UUID}.

**Response Data:**

{

“name”:” ”,

“description”:” ”,

“is\_private”:boolean,

“created\_on”:” ”,

“updated\_on”:” ”

}

**Update Project**

**Endpoints url:**

[https://api.bitbucket.org/2.0/workspaces/{workspace}/projects/{project\_key}](https://api.bitbucket.org/2.0/workspaces/%7bworkspace%7d/projects/%7bproject_key%7d)

**Http method:** PUT

**Path parameters:**

workspace: workspace id or workspace uuid.

project\_key: project key of the project.

**Request header:**

Authentication: Basic Authentication base64

Content type: application/json

**Request body:**

{

"name":" ",

"key":" ",

"description":"",

"links":{

"avtar":{

"href:<http://i.imgur.com/72tRx4w.gif>

}

},

"is\_private":false,

"created\_on":" ",

"updated\_on":" "

}

**Response Data:**

{

“name”:” ”,

“description”:” ”,

“is\_private”:boolean,

“created\_on”:” ”,

“updated\_on”:” ”

}

**Get project**

**Endpoints url:**

[https://api.bitbucket.org/2.0/workspaces/{workspace}/projects/{project\_key}](https://api.bitbucket.org/2.0/workspaces/%7bworkspace%7d/projects/%7bproject_key%7d)

**Http method:** GET

**Path parameters:**

workspace: workspace id or workspace uuid.

project\_key: project key of the project.

**Response Data:**

{

“name”:” ”,

“description”:” ”,

“is\_private”:boolean,

“created\_on”:” ”,

“updated\_on”:” ”

}

**Delete project**

**Endpoints url:**

[https://api.bitbucket.org/2.0/workspaces/{workspace}/projects/{project\_key}](https://api.bitbucket.org/2.0/workspaces/%7bworkspace%7d/projects/%7bproject_key%7d)

**Http method:** DELETE

**Path parameters:**

workspace: workspace id or workspace uuid.

project\_key: project key of the project.

**Response Data:**

Return success message.

**Create repository**

**Endpoints url:**

[https://api.bitbucket.org/2.0/repositories/{workspace}/{repo\_slug}](https://api.bitbucket.org/2.0/repositories/%7bworkspace%7d/%7brepo_slug%7d)

**Http method:** POST

**Path parameter:**

workspace: workspace id or workspace uuid

repo\_slug: the repository slug or the UUID of the repository

**Request header:**

Authentication: Basic authentication.

Content type: application/json

**Request body:**

{

"name":"",

"is\_private":true,

"readme\_type":"tutorial",

"description":"",

"forking":"no\_public\_forks",

"no\_forks":false,

"no\_public\_forks":true,

"language":"Java"

}

Note: In order to set the project for the newly created repository, pass in either the project key or the project UUID as part of the request body as shown in the examples below:

$ curl -X POST -H "Content-Type: application/json" -d '{

"scm": "git",

"project": {

"key": "MARS"

}

}' https://api.bitbucket.org/2.0/repositories/teamsinspace/hablanding

or

$ curl -X POST -H "Content-Type: application/json" -d '{

"scm": "git",

"project": {

"key": "{ba516952-992a-4c2d-acbd-17d502922f96}"

}

}' https://api.bitbucket.org/2.0/repositories/teamsinspace/hablanding

The project must only be assigned for repositories belonging to a team. If the repository owner is a team and the project is not provided, the repository is automatically assigned to the oldest project in the team.

Note: In the examples above, the username teamsinspace, and/or the repository name hablanding can be replaced by UUIDs.

**Response data:**

{

“name”:” ”,

“description”:” ”,

“created\_on”:” ”,

“updated\_on”:” ”,

“size”: ,

“language”:” “,

“has\_issues”: Boolean,

“has\_wiki”: Boolean,

“fork\_policy”:[“allow fork”,” no fork policy”,”no forks”]

}

**Get repository**

**Endpoints url:**

[https://api.bitbucket.org/2.0/repositories/{workspace}/{repo\_slug}](https://api.bitbucket.org/2.0/repositories/%7bworkspace%7d/%7brepo_slug%7d)

**Http method:** GET

**Path parameter:**

workspace: workspace id or workspace uuid

repo\_slug: the repository slug or the UUID of the repository

**Response data:**

{

“name”:” ”,

“description”:” ”,

“created\_on”:” ”,

“updated\_on”:” ”,

“size”: ,

“language”:” “,

“has\_issues”: Boolean,

“has\_wiki”: Boolean,

“fork\_policy”:[“allow fork”,” no fork policy”,”no forks”]

}

**Update repository**

**Endpoints url:**

[https://api.bitbucket.org/2.0/repositories/{workspace}/{repo\_slug}](https://api.bitbucket.org/2.0/repositories/%7bworkspace%7d/%7brepo_slug%7d)

**Http method:** PUT

**Path parameter:**

workspace: workspace id or workspace uuid.

repo\_slug: the repository slug or the UUID of the repository.

**Request header:**

Authentication: Basic authentication base64

Content type: application/json

**Request body:**

{

"name":"",

"is\_private":true,

"readme\_type":"tutorial",

"description":"",

"forking":"no\_public\_forks",

"no\_forks":false,

"no\_public\_forks":true,

"language":"Java"

}

Since this endpoint can be used to both update and to create a repository, the request body depends on the intent.

Creation

See the POST documentation for the repository endpoint for an example of the request body.

Update

Note: Changing the name of the repository will cause the location to be changed. This is because the URL of the repo is derived from the name (a process called slugification). In such a scenario, it is possible for the request to fail if the newly created slug conflicts with an existing repository's slug. But if there is no conflict, the new location will be returned in the Location header of the response.

**Response data:**

{

“name”:” ”,

“description”:” ”,

“created\_on”:” ”,

“updated\_on”:” ”,

“size”: ,

“language”:” “,

“has\_issues”: Boolean,

“has\_wiki”: Boolean,

“fork\_policy”:[“allow fork”,” no fork policy”,”no forks”]

}

**Delete repository**

**Endpoints url:**

[https://api.bitbucket.org/2.0/repositories/{workspace}/{repo\_slug}](https://api.bitbucket.org/2.0/repositories/%7bworkspace%7d/%7brepo_slug%7d)

**Http method:** DELETE

**Path parameter:**

workspace: workspace id or workspace uuid.

repo\_slug: the repository slug or the UUID of the repository.

**Required scope:**

Reposiotory: delete- Delete repositories.

**Query parameter:**

redirect\_to: If a repository has been moved to a new location, use this parameter to show users a friendly message in the Bitbucket UI that the repository has moved to a new location. However, a GET to this endpoint will still return a 404.

**Response data:**

Here there is no response from Bitbucket we can return our own response object.

**Create branch**

**Endpoints url:**

[https://api.bitbucket.org/2.0/repositories/{workspace}/{repo\_slug}/refs/branches](https://api.bitbucket.org/2.0/repositories/%7bworkspace%7d/%7brepo_slug%7d/refs/branches)

**Http method:** POST

**Path parameter:**

workspace: workspace id or workspace uuid.

repo\_slug: the repository slug or uuid of the repository.

**Request header:**

Authentication : Basic authentication base64.

Content type: application/json.

This call requires authentication. Private repositories require the caller to authenticate with an account that has appropriate authorization.

For Git, the branch name should not include any prefixes (e.g. refs/heads). This endpoint does support using short hash prefixes for the commit hash, but it may return a 400 response if the provided prefix is ambiguous. Using a full commit hash is the preferred approach.

For Mercurial, the authenticated user making this call is the author of the new branch commit and the date is current datetime of the call.

#### Required Scopes

repository:write - Read and modify your repositories

**Curl command:**

curl https://api.bitbucket.org/2.0/repositories/seanfarley/hg/refs/branches \

-s -u seanfarley -X POST -H "Content-Type: application/json" \

-d '{

"name" : "smf/create-feature",

"target" : {

"hash" : "default",

}

}

**Request body:**

{

"name" : "smf/create-feature",

"target" : {

"hash" : "default",

}

}

**Response data:**

{

"name" : "smf/create-feature",

"target" : {

"hash" : "default",

}

}

**Get Branch**

Returns a branch object within the specified repository.

**Endpoints url:**

[https://api.bitbucket.org/2.0/repositories/{workspace}/{repo\_slug}/refs/branches/{name}](https://api.bitbucket.org/2.0/repositories/%7bworkspace%7d/%7brepo_slug%7d/refs/branches/%7bname%7d)

**Http method:** GET

**Path parameter:**

workspace: workspace id or workspace uuid.

repo\_slug: repository slug or uuid of the repository.

name: name of the branch.

**Response data:**

{

"name" : "smf/create-feature",

"target" : {

"hash" : "default",

}

}

**Delete Branch**

Delete a branch in specified repository.

The main branch not allowed to be deleted and will return a 400 response.

For Git, The branch name should not included any prefixs(e.g. refs/heads.) For Mercurial, this closes all open heads on the branch, sets the author of the commit to the authenticated caller, and changes the date to the datetime of the call.

**Endpoints url:**

[https://api.bitbucket.org/2.0/repositories/{workspace}/{repo\_slug}/refs/branches/{name}](https://api.bitbucket.org/2.0/repositories/%7bworkspace%7d/%7brepo_slug%7d/refs/branches/%7bname%7d)

**Http method:** DELETE

**Path parameter:**

workspace: workspace id or workspace uuid.

repo\_slug: repository slug or uuid of the repository.

name: name of the branch.

**Required scope:**

repository:write - Read and modify your repositories.

**Response data:**

There is no response data from bitbucket server we can return our own success message as response.