Table of Contents

[Authorization Method 2](#_Toc341208028)

[Need upgrade 2](#_Toc341208029)

[Addfolder 3](#_Toc341208030)

[Save Input parameters 3](#_Toc341208031)

[Renameobject 4](#_Toc341208032)

[Deleteobject 4](#_Toc341208033)

[Move 5](#_Toc341208034)

[List 5](#_Toc341208035)

[GET 6](#_Toc341208036)

[Allowed extension 6](#_Toc341208037)

[Update 6](#_Toc341208038)

Authorization Method**:**

**Input parameters:**

* Retrieve Username from request parameters
* Retrieve password from request parameters
* Retrieve API Key from request parameters

**Business Logic:**

Check if username is not valid or password is empty. If Yes, then

Check if key is empty then

Show message “Invalid credentials” and exit

Match the user’s key from database. Fetch the data from database and match it with the supplied values. If verified then return data else return “Invalid Credentials”

Else

Match the user’s username & password from database. Fetch the data from database and verify the username password from table

If verified then return data otherwise show message “Invalid Credentials”

# Need upgrade

**Input parameters:**

* Retrieve id from request parameters e.g. dropbox, office, outlook.
* Retrieve APIKey from parameters.
* Retrieve version from request parameters.

**Business Logic:**

* Check If application id is empty then Show Message “Please identify your request with app id.” exit
* Check If version is valid then Show message “Inappropriate version”
* Check If application version is less than latest version available then upgrade is required and return latest setup url else no upgrade is required.

# Addfolder

**Input parameters**

* Retrieve Folder Name from parameters
* Retrieve space from parameters
* Retrieve Parent Folder Id from parameters
* Retrieve APIKey from parameters.
* Retrieve ShareId from parameters

**Business Logic**

* Check if parent Folder id is Valid if No then show Message “Invalid Folder ID”.
* Check if folder name is empty. If Yes, then Show Message “Please specify folder name.” and Exit
* Check If selected space is shared and parent folder is Root then Show Message “Sorry can't create root level folders in shared space.” and exit
* Check if user has permissions to create folder, if not show message “You don’t’ have permissions to add folders.” And exit.
* Otherwise Create folder with specified folder name and show message “Folder added successfully”.

# Save Input parameters

* Retrieve Folder Name from parameters
* Retrieve space from parameters
* Retrieve Parent Folder Id from parameters
* Retrieve APIKey from parameters.
* Retrieve Filesize from parameters.

**Business Logic**

* Check if parent Folder id is Valid if No then show Message “Invalid Folder ID”.
* Check File size if filesize lessthan 0 then Show Message “Cannot upload file size is zero - btw, why are you trying to be so negative? :-P”.
* Check If selected space is shared and parent folder is root. If yes, then Show Message “Sorry can't create root level folders in shared space.” and Exit.
* Check If user has access permissions. If no, then Show Message “You don’t have acess permissions”.
* Check if user has write permissions. If Not return with proper message.
* Check the file extension is allowed or if not return with proper message.
* Check if file already exists as defined condition then overwrite the file and update version of file .
* If we have new file then use the file name and store file on target folder
* Update the name of documents
* After successfully upload update file status .
* After successfully uploading and update the file status and return with message “File saved successfully.”

# Renameobject

**Input parameters**

* Retrieve Object Id from request parameters.
* Retrieve space from parameters
* Retrieve Object Type from request parameters.Object Type – folder or file
* Retrieve APIKey from parameters.
* Retrieve new name from parameters

**Business Logic**

* Check If object id is valid. If not, then show message “Object id is invalid.” And exit.
* Check If object type is valid. If not, then Show Message “Object type is invalid.” And exit.
* Check if new name is empty. If yes, then Show Message “Object new name is invalid.” And exit.
* Check if file name is valid. If not, Show Message “Invalid Name” and exit.
* Check the permissions for space, read write permissions
* Check If user has write permission on the file. If yes, then show Message “You don't have permission to rename the object.”
* Check if object type is folder. If yes, then retrieve folder id and parent id
* Check if folder with the new name already exists. If yes, then show message “Folder already exists.” And exit
* Check if file with the same name already exists in same folder then show message “File with same name already exists” and exit.
* Update name in database and show message “successfully renamed.”

# Deleteobject

**Input parameters**

* Retrieve Object Id from request parameters.
* Retrieve space from parameters
* Retrieve Object Type from request parameters. Object Type – folder or file
* Retrieve APIKey from parameters.

**Business Logic**

* Check If object id is invalid. If yes, then Show Message “Object id is invalid/missed.” And exit
* Check If object type is invalid. If yes, then show message “Object type is invalid/missed.” And exit
* Check if Object Type is Folder. If yes then change the status of that folder and show message “Folder deleted successfully”.
* Check if Object Type is Files. if yes then change the status of that folder and show message “files deleted successfully”.
* Check if object deleted successfully. If yes, then show message “object deleted successfully”
* Otherwise show error message “Deletion failed”

# Move

**Input parameters**

* Retrieve Source folder id from request parameters.
* Retrieve space from parameters.
* Retrieve Destination folder id from request parameters.
* Retrieve APIKey from parameters.

**Business Logic**

* Retrieve source id and destination id from parameters
* Check If source id is invalid. If yes, then show message “invalid source id” and exit
* Check If destination id is invalid. If yes, then show message”invalid destination id ”.and exit
* Check user rights for move
* If the user doesn’t have rights for move then show message "No rights to move” and exit.
* Check if object type is folder. If yes, then
* Check if folder id is valid. If yes, then perform move folder operations and update database.
* Check if object type is file. If yes, then update the database
* Check If folder or file is successfully moved. If yes, then show message “Object Successfully moved”
* Otherwise show message “Unable to move”.

# 

# List

**Input parameters**

* Retrieve folder id from request parameters.
* Retrieve space from parameters.
* Retrieve Destination folder id from request parameters.
* Retrieve APIKey from parameters.
* Retrieve everthing from parameters

**Business Logic**

* Retrieve folder id from parameters.
* Retrieve the value of everything
* Check if everything is equal to 1. If yes, then
* Check the space
* Check for permissions
* Check if space is company and parent folder id is =0 then get the list of company space objects types
* Check if space is personal and parent folder id is =0 then get the list of personals space objects types
* Check if space is shared then get the list of shared space objects types
* and return list of objects in the selected folder.

# GET

**Input parameters**

* Retrieve Document id from request parameters.
* Retrieve space from parameters.
* Retrieve version from request parameters.
* Retrieve APIKey from parameters.

**Business Logic**

* Retrieve document version
* Check if selected Space is Shared and Shareid=0. If yes, then Show message “Invalid Share ID. Please specify ShareID in your request.” And exit
* find file name and versionnumber and user\_id .
* Check If file not found. If not found then return false. exit
* Use document name , file name , and file location for file retrieval and read file
* Otherwise show message “Error! File not found”.

# Allowed extension

**Input parameters**

* Retrieve APIKey from parameters.

**Business Logic**

* Returns all the allowed extension from the config file .

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_