CSE 5382 **Secure Programming Fall 2018, Assignment 3**

Swetha, Vijaya Raghavan | sxv1229| 1001551229

Implemented the following functionalities:

SP Share authenticates each user and Multiple concurrent users can share "items" such as pictures, text, executables (other format "binary" (raw) files), small videos.
There are three types of SPS "users": general, admin and root.
Root will grant all permissions to admin, and admin can approve any user.

General user will be registered with the actual name other than login name along with password and the group name.

All the General users who have been authenticated may View, Put, Get, and Remove, (as well as edit) items: pictures, videos, text, and other format "binary" (raw) files.

View – selects all the items in all groups that a user is a member of and display them to the user.

Get – This functionality will extract the item (meta data of the files along with its file id(if flat-null else fileid), item id, timestamp, item description of all the files the user has uploaded).

Remove – Drops the entire item entry from mysql database. This privilege is given only for the item creator or the admin.

Put – Inserts the data into the respective item table in the mysql database. (item description as creator name, timestamp etc)

Download – Downloads the file for the file id mentioned to the user.

Group names and general users must be accepted by the admin before being activated (asynchronously).

The admin must accept (allow) user (who have previously registered), requested groups, be able to create new groups, and be able to set some limits of space consumption: limits per individual user, limits per group, and limits per item.

Users (as well as admin) may operate independently (asynchronously).
Items (group files) are stored in a mysql database.

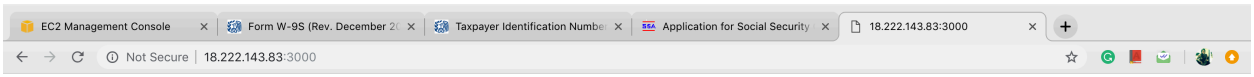
All passwords, protection, etc must be "secure". – We are hashing the password and generating a token through which the user is authenticated the next time he logs in.

All the users will be signedup(registered) with their unique loginname, password and actual name.

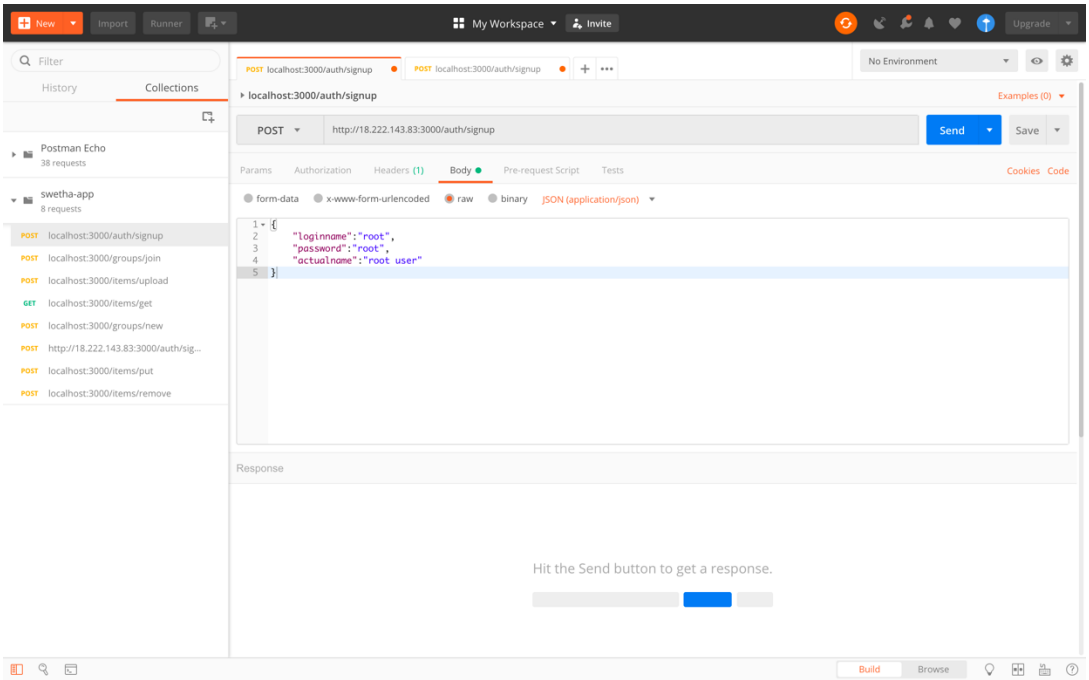
All the other mentioned description for each filed names is taken into consideration.

Deployed in AWS. http://18.222.143.83:3000/

References: StackOverFlow, Wikipedia



App is Live .....

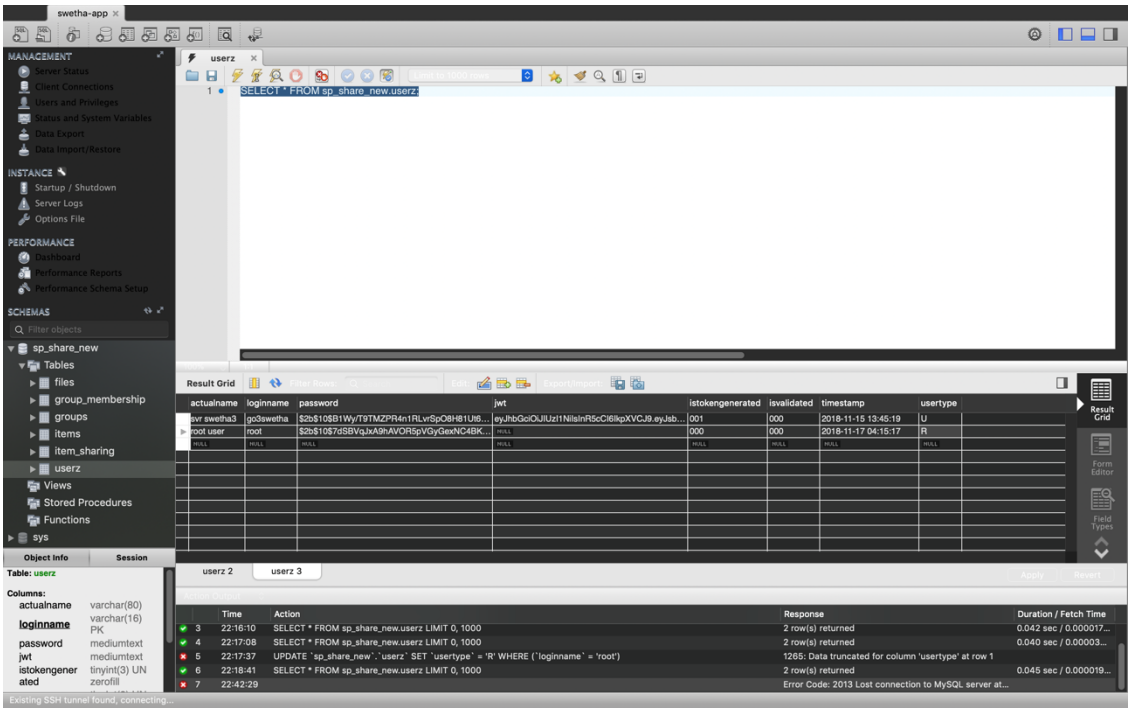


```
    router.post('/signup', function (req, res) {
        console.log(req);
        let saltRounds = 10;
        let originalpwd = connection.escape(req.body.password);
        let loginname = connection.escape(req.body.loginname);
        let actualname = connection.escape(req.body.actualname);

        if (loginname !== null) {
            connection.query("SELECT * FROM sp_share_new.userz userz WHERE upper(userz.loginname) = upper(" + loginname + ")", function (err, result, fields) {
                if (result.length == 0) {
                    bcrypt.hash(originalpwd, saltRounds, function (error, hashedPwd) {
                        let hashed_Pwd = connection.escape(hashedPwd)
                        connection.query("Insert into sp_share_new.userz (actualname, loginname, password) VALUES (" + actualname + "," + loginname + "," + hashed_Pwd + ")",
                            if (error) {
                                res.sendStatus(409);
                            } else {
                                res.sendStatus(200);
                                usage_monitor.initUserUsage(loginname);
                            }

                        });
                    });
                } else {
                    res.sendStatus(409);
                }

            });
        }
    });
```

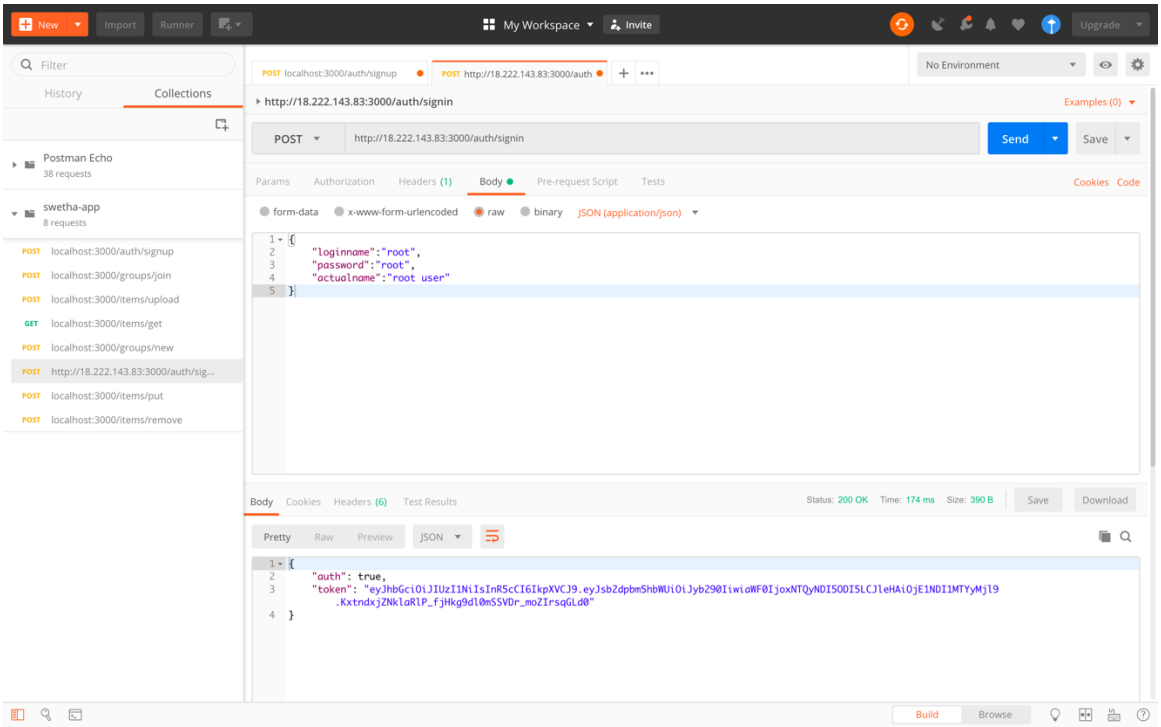
The password sent will be hashed based on salts(number of salts used is 10). Used bcrypt method. Also created a jwt token which expires in 24 hours. This will be updated only when is_activated is true(only for active users). Attached the method screenshot below.

```
module.exports = function (connection, bcrypt, jwt, config, shortid) {
    router.post('/signin', function (req, response) {
        let password = req.body.password;
        let loginname = req.body.loginname;
        if (loginname !== null && password !== null) {
            connection.query("SELECT password FROM sp_share_new.userz WHERE loginname = " + connection.escape(loginname), function (err, result) {
                console.log(result);
                if (result) {
                    var hash = result[0]['password'];
                    bcrypt.compare(password, hash, function (err, res) {
                        console.log(res);
                        if (res) {
                            // create a token
                            var token = jwt.sign({
                                    loginname: loginname
                                }, config.secret, {
                                    expiresIn: 86400 // expires in 24 hours
                                },
                                function (err, token) {
                                    connection.query("UPDATE sp_share_new.userz SET jwt = " + connection.escape(token) + ", istokengenerated = true WHERE loginname = " + conne
                                    response.status(200).send({
                                        auth: true,
                                        token: token
                                    });
                                    console.dir(this);
                                }
                            );

                        }
                    });
                }
            });
        }

    });
```

Based on the loginname, if it is root, the user type will be “R”(root) else it will be “U”(general user) or “A”(admin). In Database, under userz table, an entry will be created. Sample screenshot attached below.



Once the user is signed up, the same user can be logged in to the system. In the screenshot below, you can see we can give loginname, password, actual name is optional here and, in the response, we will get a token generated only when the authentication is done.



Once we get the bearer token, I will match the login name obtained and stored loginname from db and if it matches then the user is authenticated else Failed to authenticate token message is displayed.

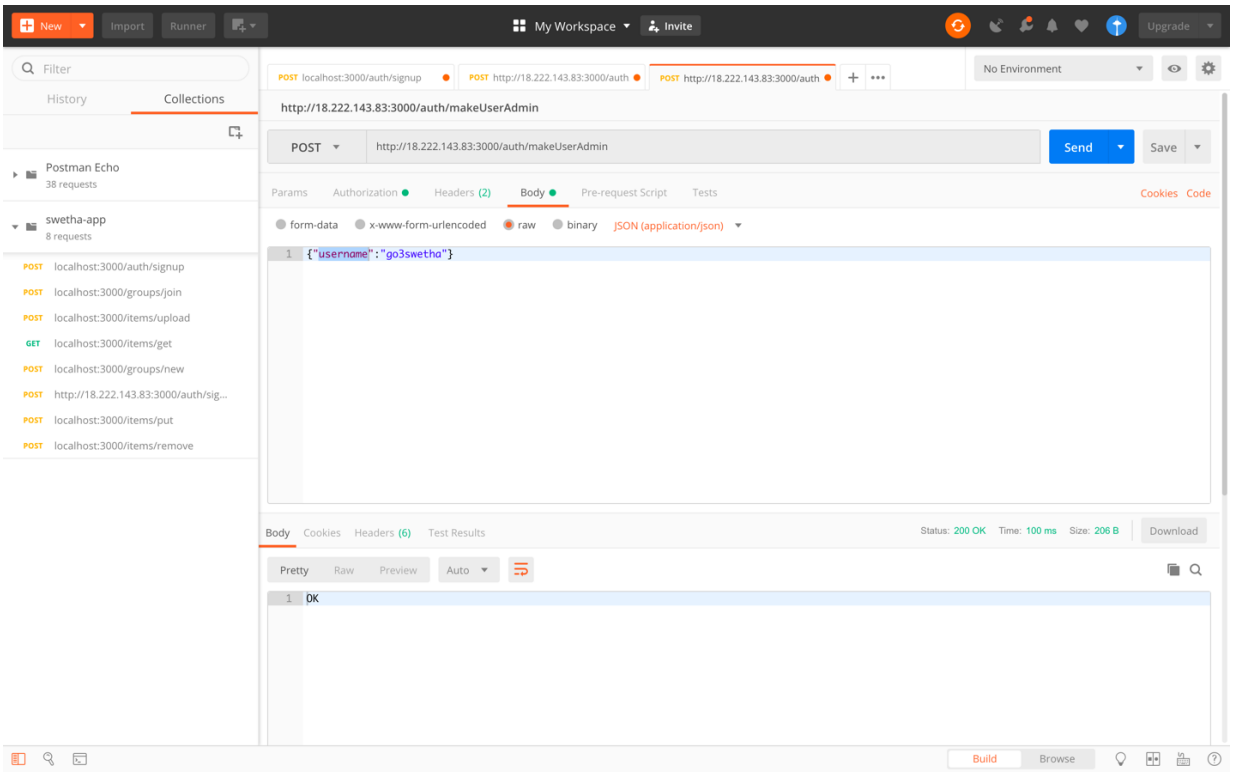
BasicAuthValidation.js

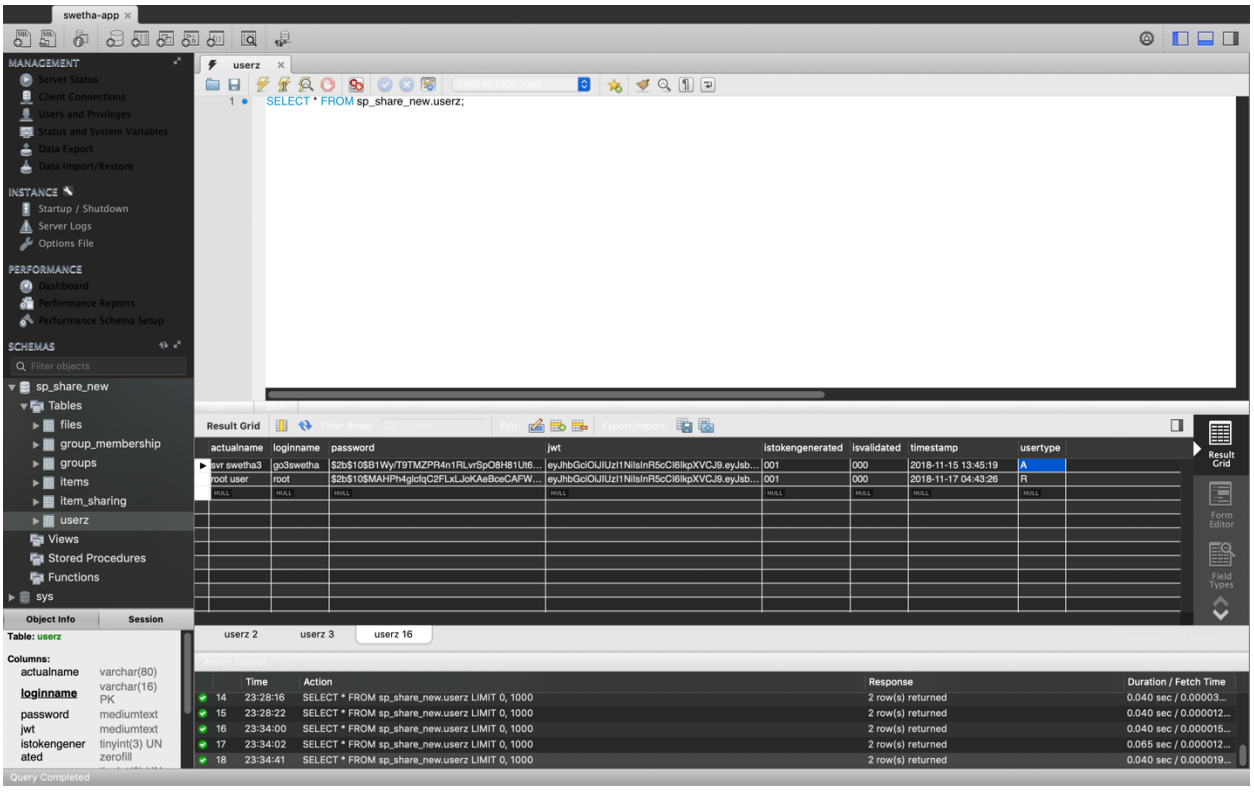
```
module.exports = function (connection, bcrypt, jwt, config, token,req,res) {
    return function(callback){
        if (!token) return res.status(401).send({
            auth: false,
            message: 'No token provided.'
        });
        jwt.verify(token, config.secret, function (err, decoded) {
            if (err) return res.status(500).send({
                auth: false,
                message: 'Failed to authenticate token.'
            });
            if (!err) {
                connection.query("SELECT loginname FROM sp_share_new.userz WHERE loginname = " + connection.escape(decoded.loginname) + " AND jwt = " + connection.escape(toke
                    if (result) {
                        if(typeof callback == 'function'){
                            req.local = {};
                            req.local.loginname = decoded.loginname;
                            // res.local = {};
                            // res.local.loginname = decoded.loginname;
                            callback();
                        }
                    }
                });
            }
        });


    }
}
```

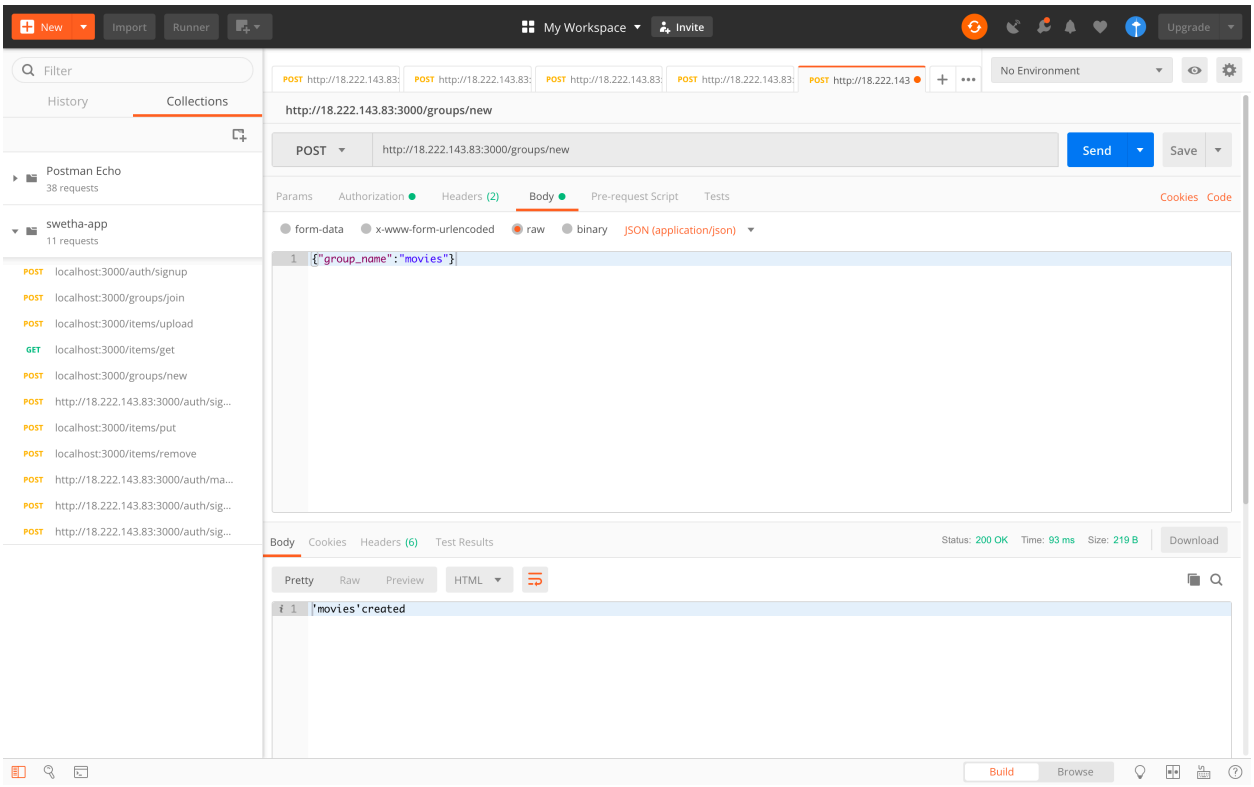
Line 20, Column 40 Spaces: 4 JavaScript

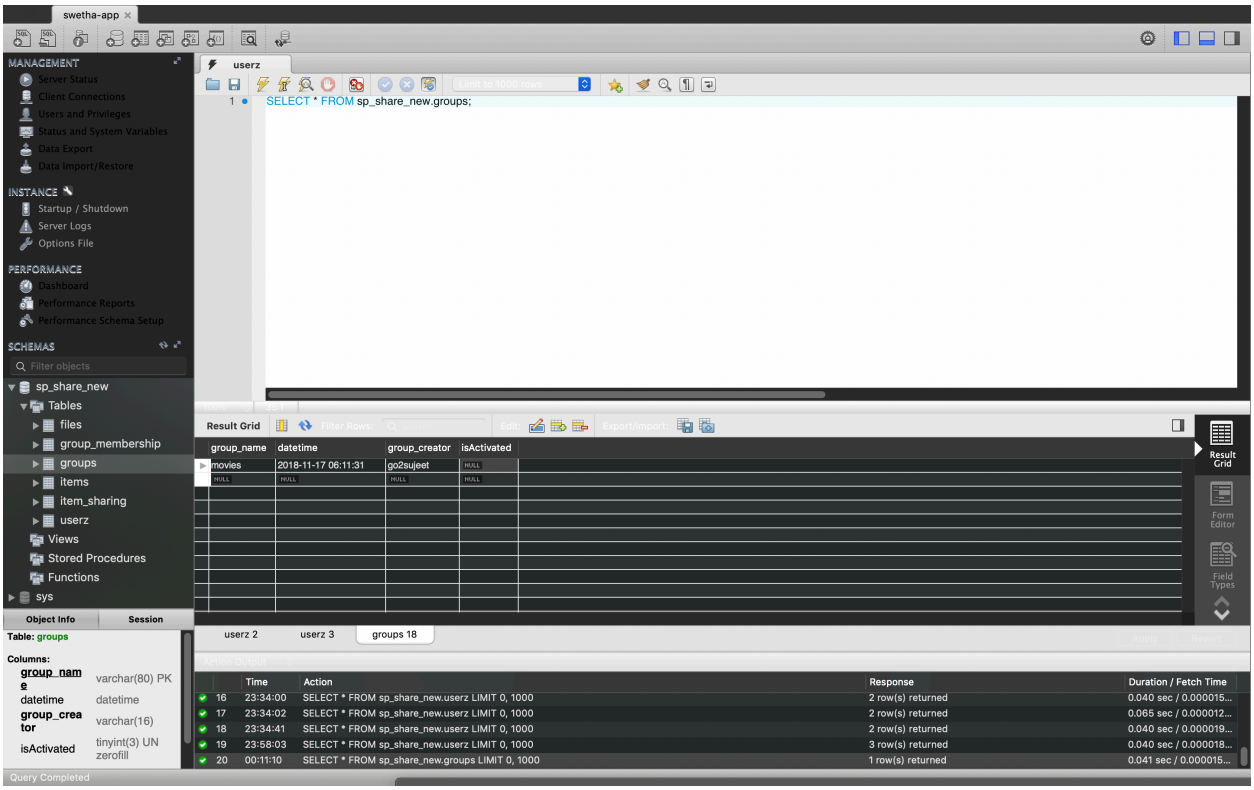
Once the user is authenticated, root can make any user as admin described in makeUserAdmin, and in the db, we can see first `go3swetha` had a user type `U` now reflected to `A`.

```
    router.post('/makeUserAdmin', function (req, res) {
        let username = connection.escape(req.body.username);
        let xx = req.header('authorization');
        let TokenArray = xx.split(" ");
        let token = TokenArray[1];
        let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
        _bav(function () {
            let login_name = connection.escape(req.local.loginname);
            if (String(req.body.username).toLowerCase() == 'root') {
                res.sendStatus(401);
            } else {
                connection.query("select userz.loginname from userz where userz.usertype='R' and userz.loginname =" + login_name, function (error, result) {
                    if (!error && result.length > 0) {
                        connection.query("UPDATE `sp_share_new`.`userz` SET `usertype`='A' WHERE  `loginname`=" + username, function (error, result) {
                            if (!error) {
                                res.send(200)
                            }
                        })
                    }
                })
            }

        })
    })
    return router;
}
```


New
Import
Runner
My Workspace
Invite
Upgrade
Filter
History
Collections
Postman Echo
38 requests
swetha-app
8 requests
POST localhost:3000/auth/signup
POST localhost:3000/groups/join
POST localhost:3000/items/upload
GET localhost:3000/items/get
POST localhost:3000/groups/new
POST http://18.222.143.83:3000/auth/sig...
POST localhost:3000/items/put
POST localhost:3000/items/remove
POST localhost:3000/auth/signup
POST http://18.222.143.83:3000/auth
POST http://18.222.143.83:3000/auth
No Environment
http://18.222.143.83:3000/auth/makeUserAdmin
POST
http://18.222.143.83:3000/auth/makeUserAdmin
Send
Save
Params
Authorization
Headers (2)
Body
Pre-request Script
Tests
Cookies
Code
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
1 {"username":"go3swetha"}
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 100 ms
Size: 206 B
Download
Pretty
Raw
Preview
Auto
1 OK
Build
Browse


swetha-app
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
PERFORMANCE
Dashboard
Performance Reports
Performance Schema Setup
SCHEMAS
Filter objects
sp_share_new
Tables
files
group_membership
groups
items
item_sharing
userz
Views
Stored Procedures
Functions
sys
Object Info
Session
Table: userz
Columns:
actualname varchar(80)
loginname varchar(16) PK
password mediumtext
jwt mediumtext
istokengener ated tinyint(3) UN zerofill
userz
1 SELECT * FROM sp_share_new.userz;
Result Grid
actualname loginname password jwt istokengenerated isvalidated timestamp usertype
avr swetha3 go3swetha $2b$10$B1WyT9TMZPR4n1RLvrSpO8H81Ut6... eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJsb... 001 000 2018-11-15 13:45:19 A
root user root $2b$10$MAHPh4glcfqC2FLxLJoKAeBceCAFW... eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJsb... 001 000 2018-11-17 04:43:26 R
userz 2
userz 3
userz 16
Time Action Response Duration / Fetch Time
14 23:28:16 SELECT * FROM sp_share_new.userz LIMIT 0, 1000 2 row(s) returned 0.040 sec / 0.00003...
15 23:28:22 SELECT * FROM sp_share_new.userz LIMIT 0, 1000 2 row(s) returned 0.040 sec / 0.000012...
16 23:34:00 SELECT * FROM sp_share_new.userz LIMIT 0, 1000 2 row(s) returned 0.040 sec / 0.000015...
17 23:34:02 SELECT * FROM sp_share_new.userz LIMIT 0, 1000 2 row(s) returned 0.065 sec / 0.000012...
18 23:34:41 SELECT * FROM sp_share_new.userz LIMIT 0, 1000 2 row(s) returned 0.040 sec / 0.000019...
Query Completed

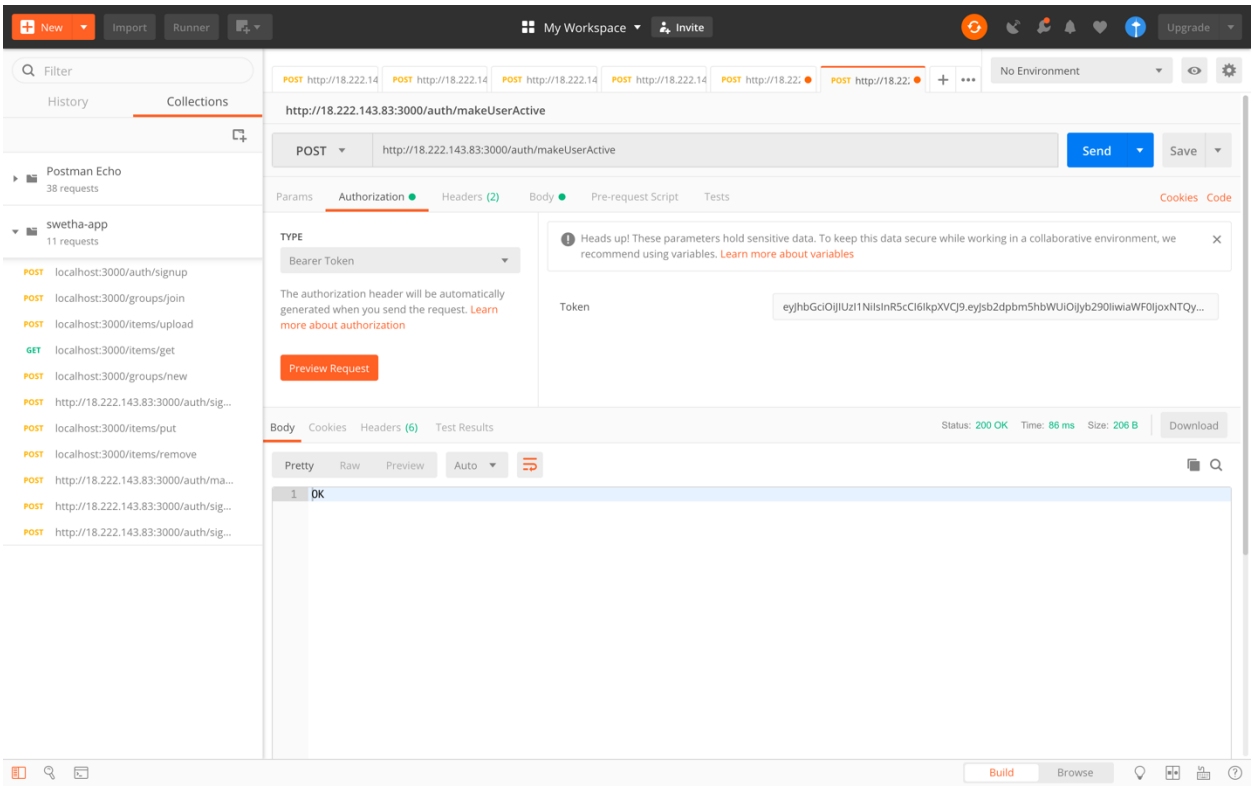

New
Import
Runner
My Workspace
Invite
Upgrade
Filter
History
Collections
Postman Echo
38 requests
swetha-app
11 requests
POST localhost:3000/auth/signup
POST localhost:3000/groups/join
POST localhost:3000/items/upload
GET localhost:3000/items/get
POST localhost:3000/groups/new
POST http://18.222.143.83:3000/auth/sig...
POST localhost:3000/items/put
POST localhost:3000/items/remove
POST http://18.222.143.83:3000/auth/ma...
POST http://18.222.143.83:3000/auth/sig...
POST http://18.222.143.83:3000/auth/sig...
No Environment
http://18.222.143.83:3000/groups/new
POST
http://18.222.143.83:3000/groups/new
Send
Save
Params
Authorization
Headers (2)
Body
Pre-request Script
Tests
Cookies
Code
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
{"group_name":"movies"}
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 93 ms
Size: 219 B
Download
Pretty
Raw
Preview
HTML
'movies'created
Build
Browse


swetha-app
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
PERFORMANCE
Dashboard
Performance Reports
Performance Schema Setup
SCHEMAS
Filter objects
sp_share_new
Tables
files
group_membership
groups
items
item_sharing
userz
Views
Stored Procedures
Functions
sys
Object Info
Session
Table: groups
Columns:
group_name varchar(80) PK
datetime datetime
group_creator varchar(16)
isActivated tinyint(3) UN zerofill
userz
SELECT * FROM sp_share_new.groups;
Result Grid
group_name datetime group_creator isActivated
movies 2018-11-17 06:11:31 go2sujeet
userz 2
userz 3
groups 18
Action Output
Time Action Response Duration / Fetch Time
16 23:34:00 SELECT * FROM sp_share_new.userz LIMIT 0, 1000 2 row(s) returned 0.040 sec / 0.000015...
17 23:34:02 SELECT * FROM sp_share_new.userz LIMIT 0, 1000 2 row(s) returned 0.065 sec / 0.000012...
18 23:34:41 SELECT * FROM sp_share_new.userz LIMIT 0, 1000 2 row(s) returned 0.040 sec / 0.000019...
19 23:58:03 SELECT * FROM sp_share_new.userz LIMIT 0, 1000 3 row(s) returned 0.040 sec / 0.000018...
20 00:11:10 SELECT * FROM sp_share_new.groups LIMIT 0, 1000 1 row(s) returned 0.041 sec / 0.000015...
Query Completed

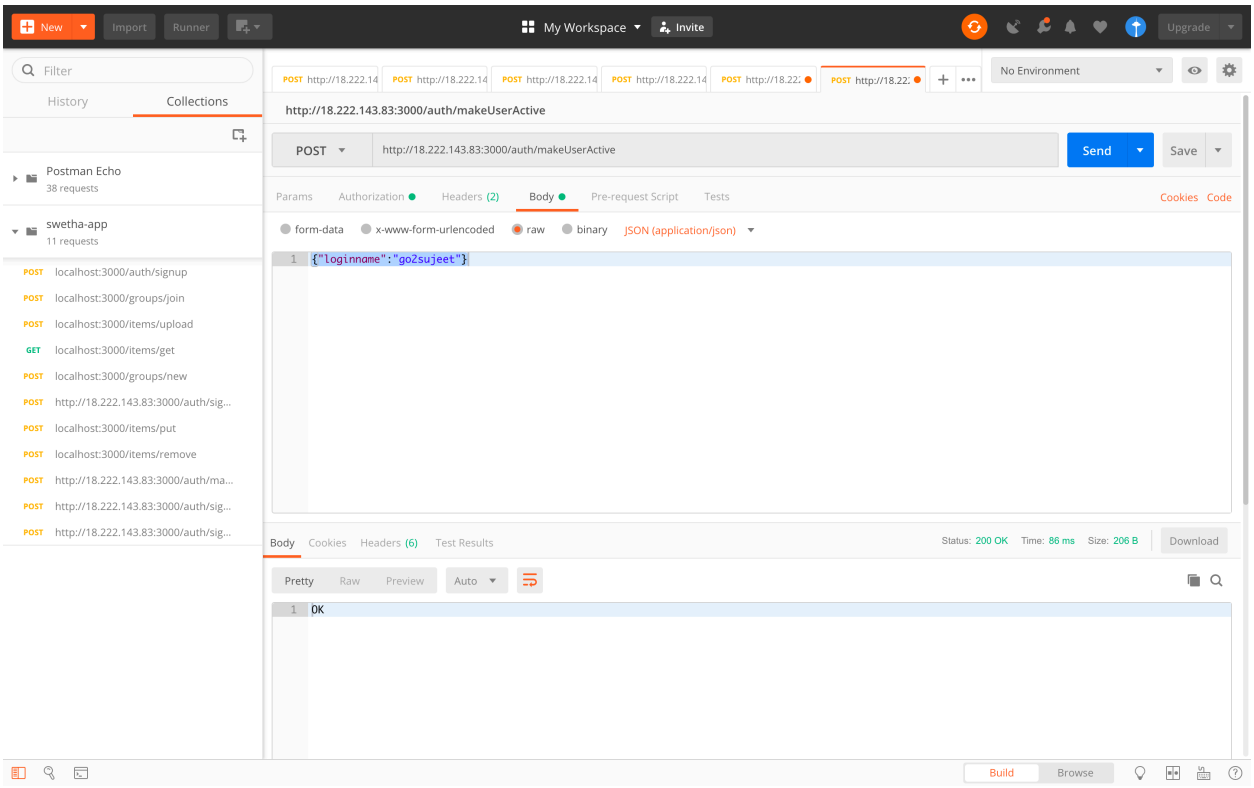
```
router.post('/new', function (req, res, next) {
    var xx = req.header('authorization');
    let TokenArray = xx.split(" ");
    var token = TokenArray[1];
    const _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
    _bav(function (loginname) {
        next();
    })
}, function (req, res, next) {
    let group_name = connection.escape(req.body.group_name);
    let loginname = connection.escape(req.local.loginname);
    connection.query("INSERT INTO sp_share_new.`groups`(group_name, group_creator) VALUES (" + group_name + ", " + loginname + ")", function (err, result) {
        if (!err) {
            res.status(200).send(group_name + 'created');
            usage_monitor.initGroupUsage(group_name);

        } else{
            res.status(406).send(group_name + ': cannot be created');
        }
    });

})
```

The Root will activate the admin, in turn admin will activate the user (The admin must accept (allow) user (who have previously registered), requested groups)

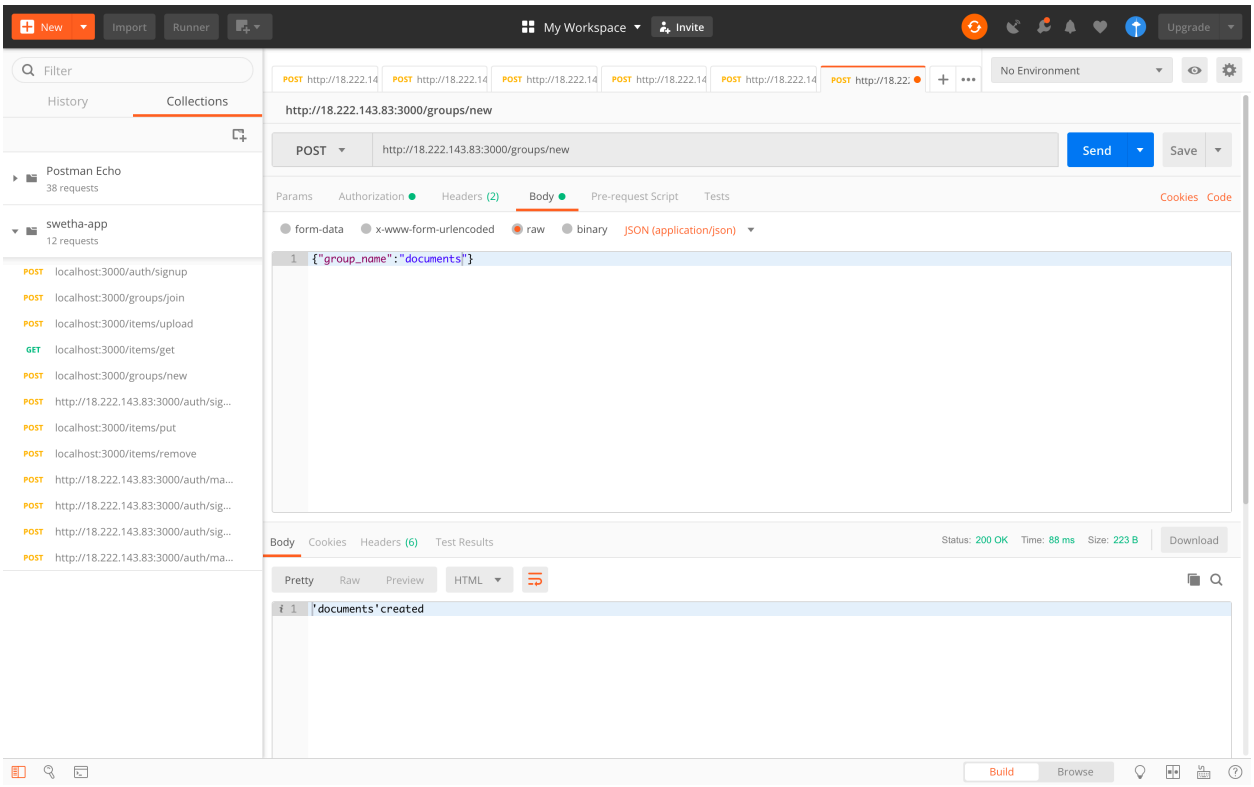
Who ever is activating the general user, their key will be provided in the bearer token and loginname will be the new user who is yet to get activated. This is implemented under makeUserActive route.

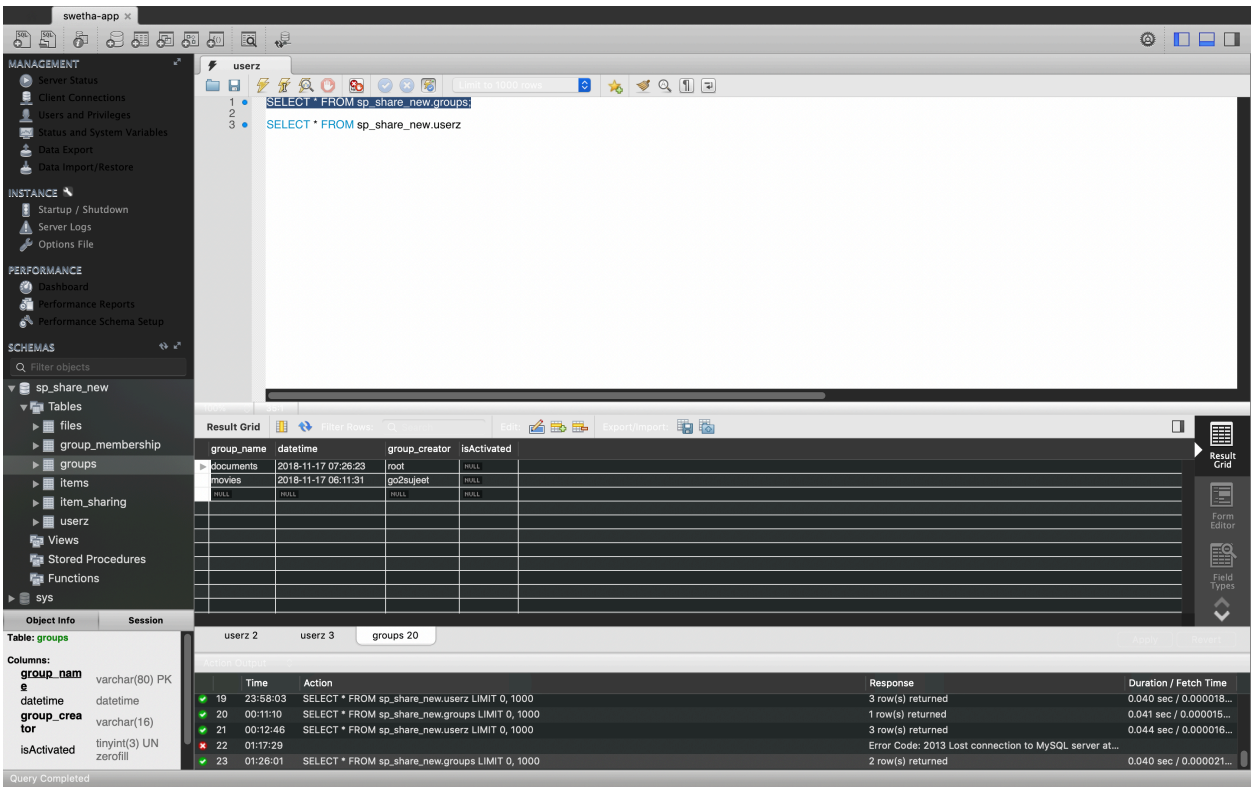


```
    router.post('/makeUserAdmin', function (req, res) {
        let username = connection.escape(req.body.username);
        let xx = req.header('authorization');
        let TokenArray = xx.split(" ");
        let token = TokenArray[1];
        let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
        _bav(function () {
            let login_name = connection.escape(req.local.loginname);
            if (String(req.body.username).toLowerCase() == 'root') {
                res.sendStatus(401);
            } else {
                connection.query("select userz.loginname from userz where userz.usertype='R' and userz.loginname =" + login_name, function (error, result) {
                    if (!error && result.length > 0) {
                        connection.query("UPDATE `sp_share_new`.`userz` SET `usertype`='A' WHERE  `loginname`=" + username, function (error, result) {
                            if (!error) {
                                res.send(200)
                            }
                        })
                    }
                })
            }

        })
    })
    return router;
}
```

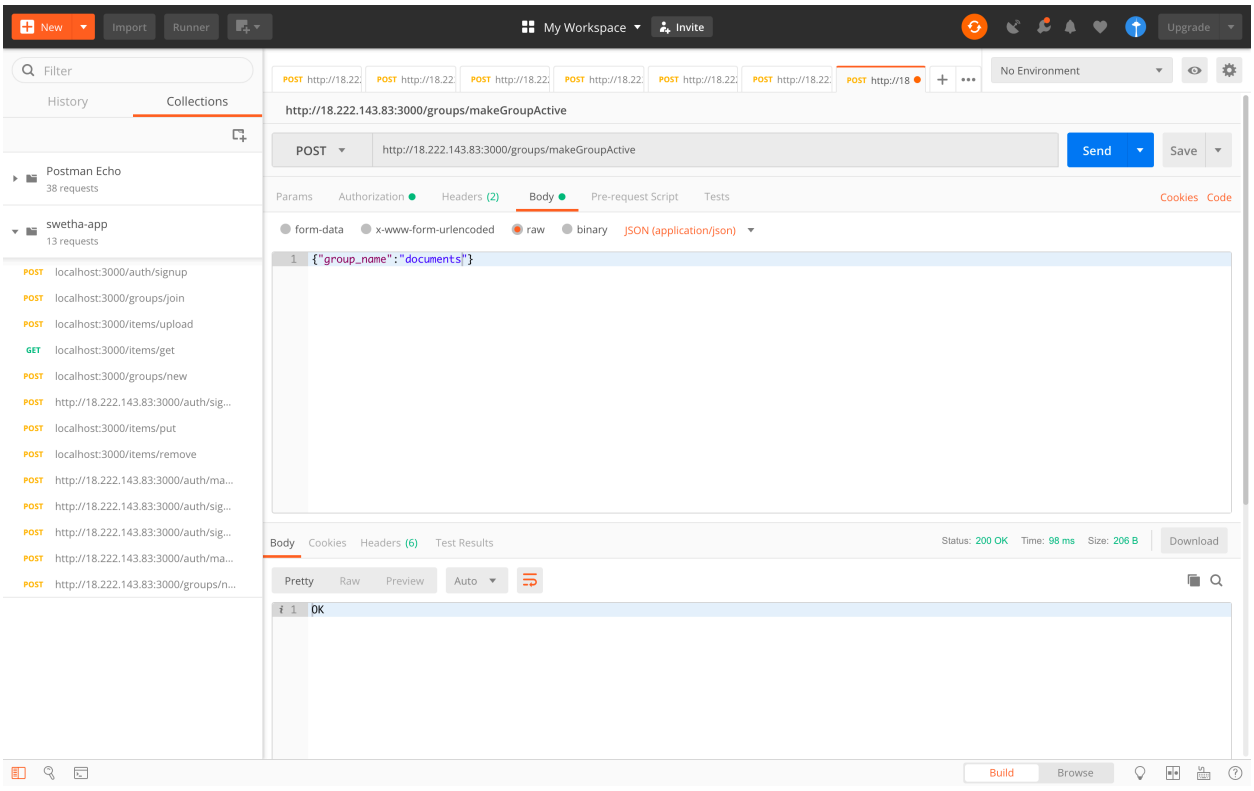


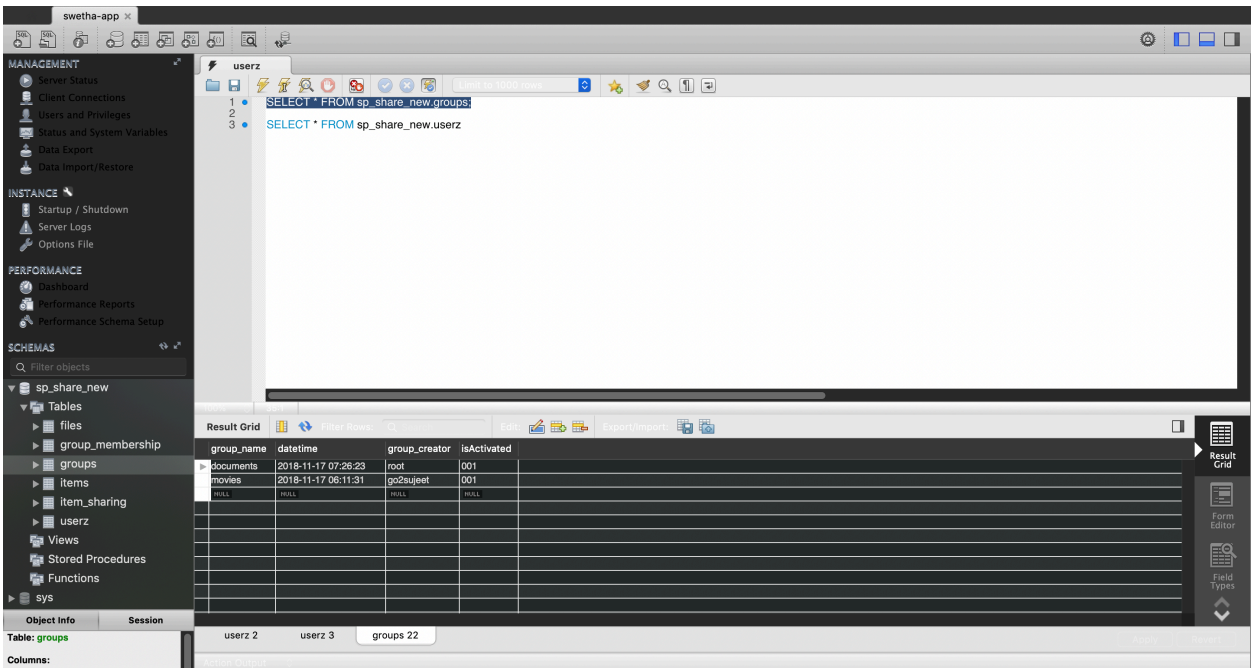
The group is created, and it will be stored in groups table in the db



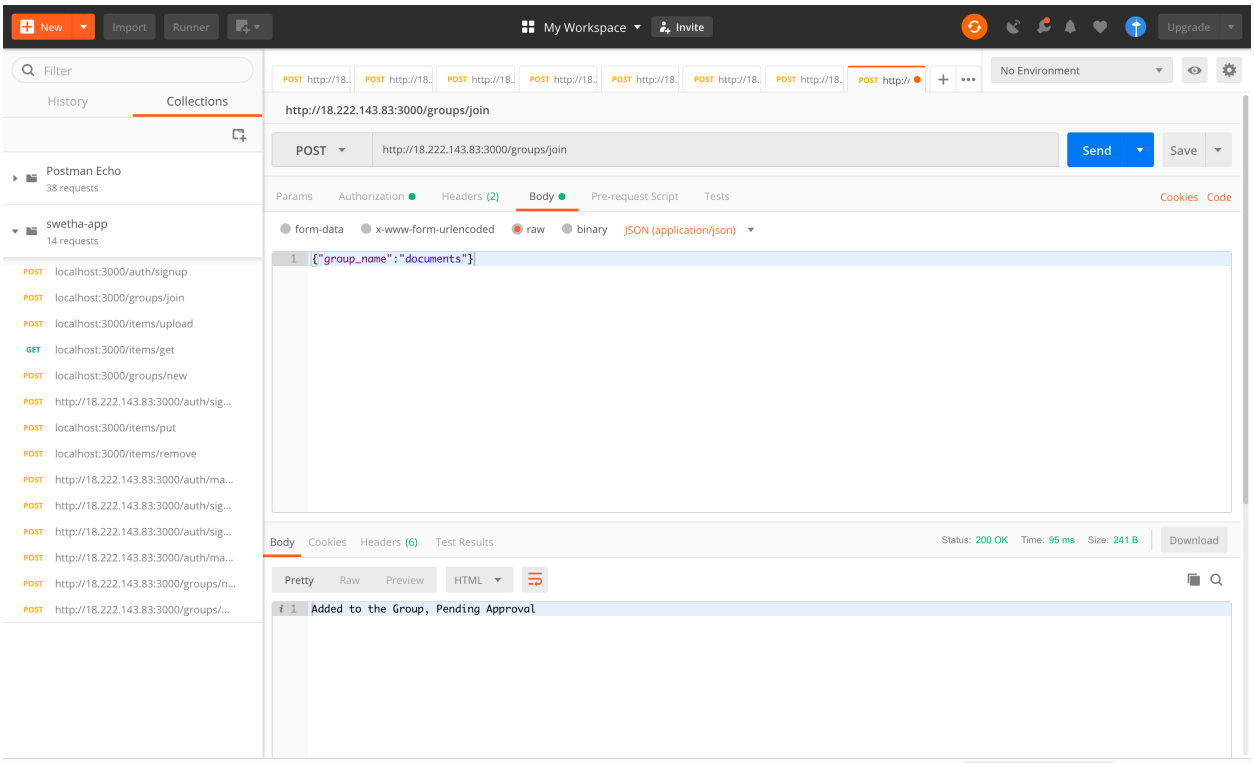


Once the group is created, the group has to be approved by the root or admin.

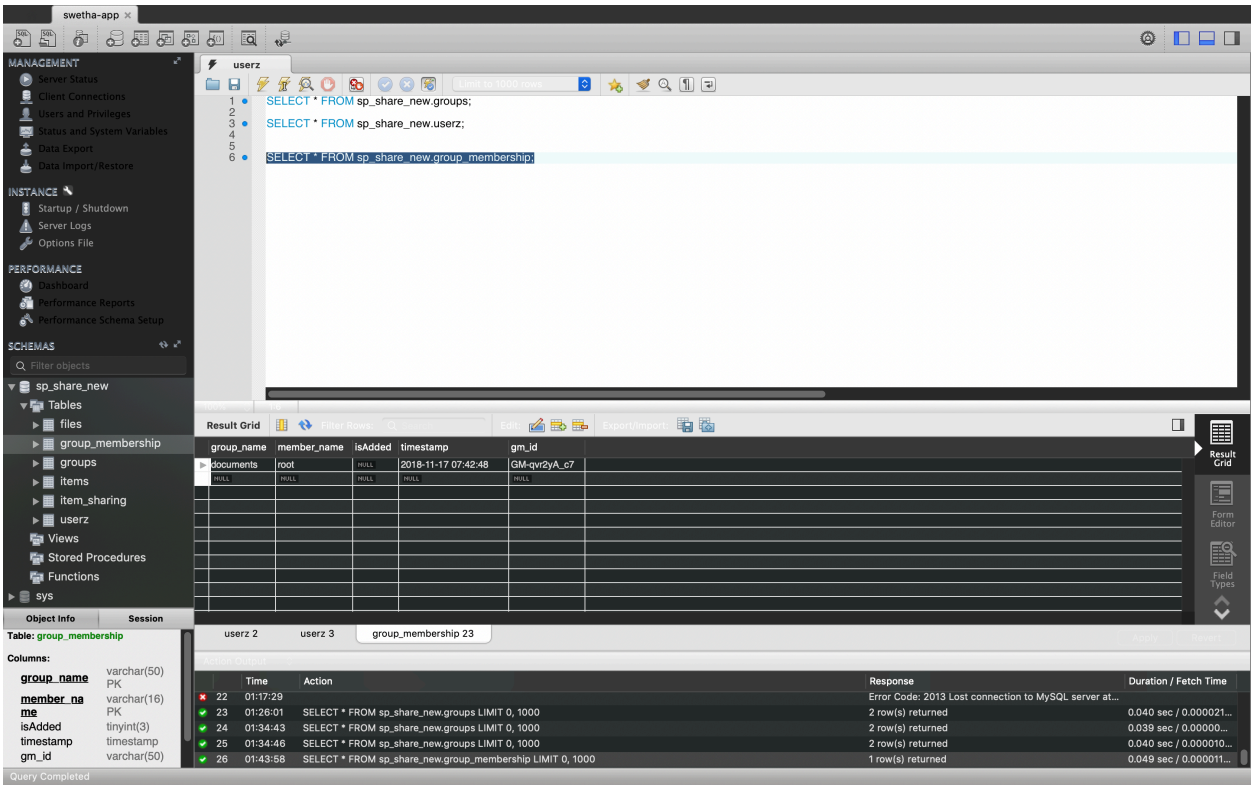




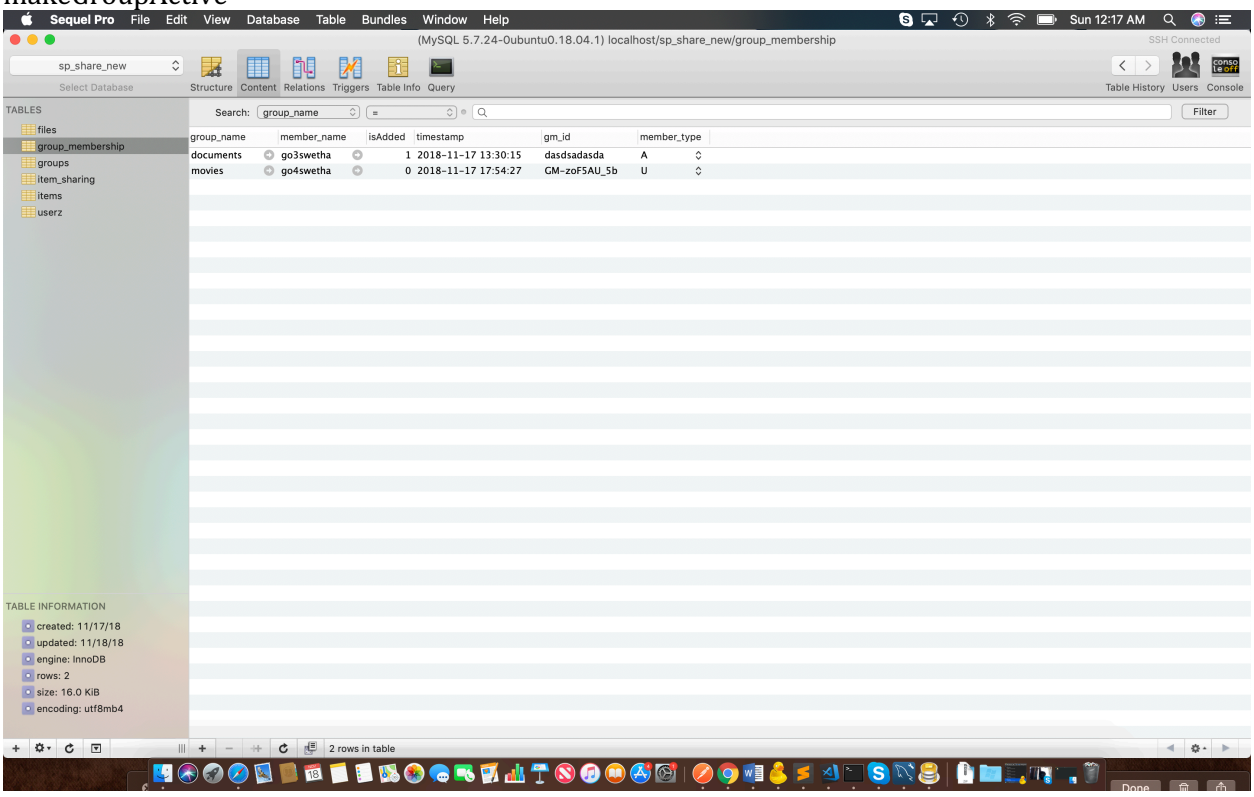
Eventhough user has created the group, not necessarily he needs to join. So I have an route which joins user to the provided `group_name`. This needs admin or root approval.

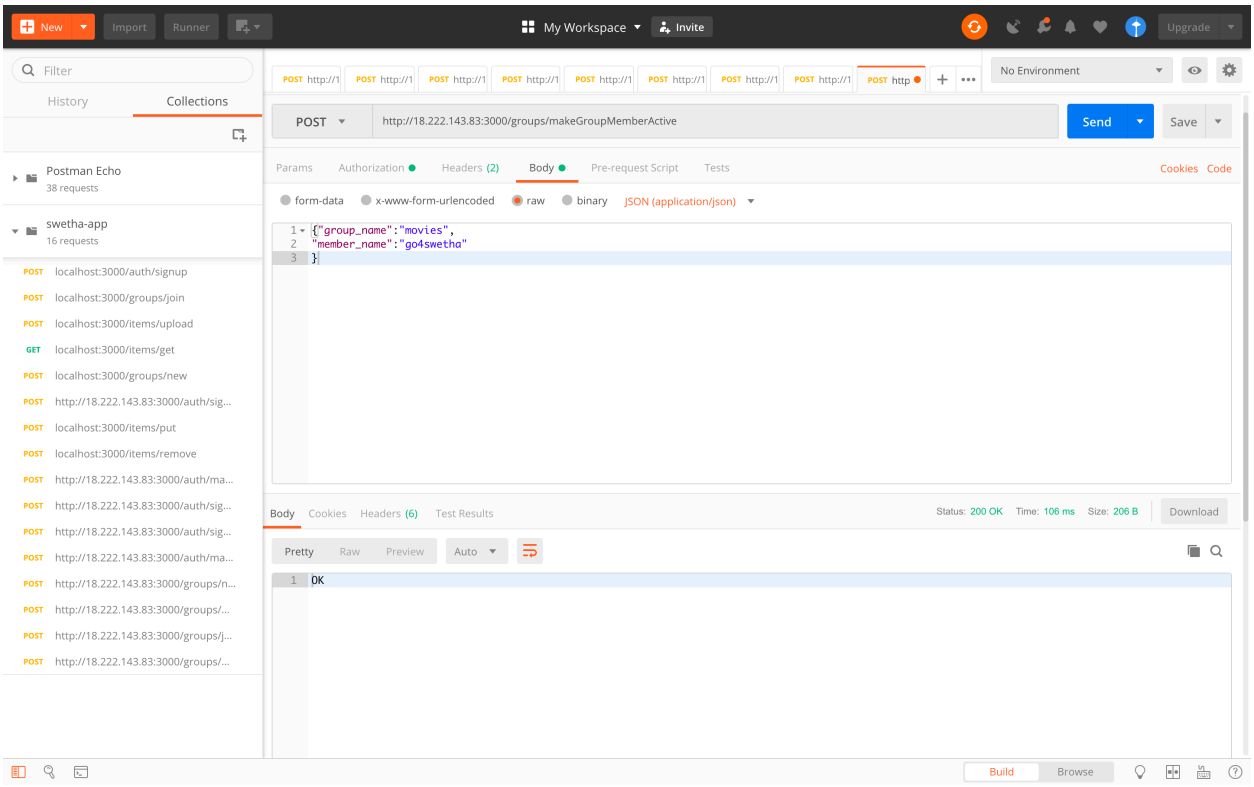


Once user is requested to join a specific group by providing a group_name, this will be store in group_membership table. Since the user is not yet approved, the isAdded field is by default false.

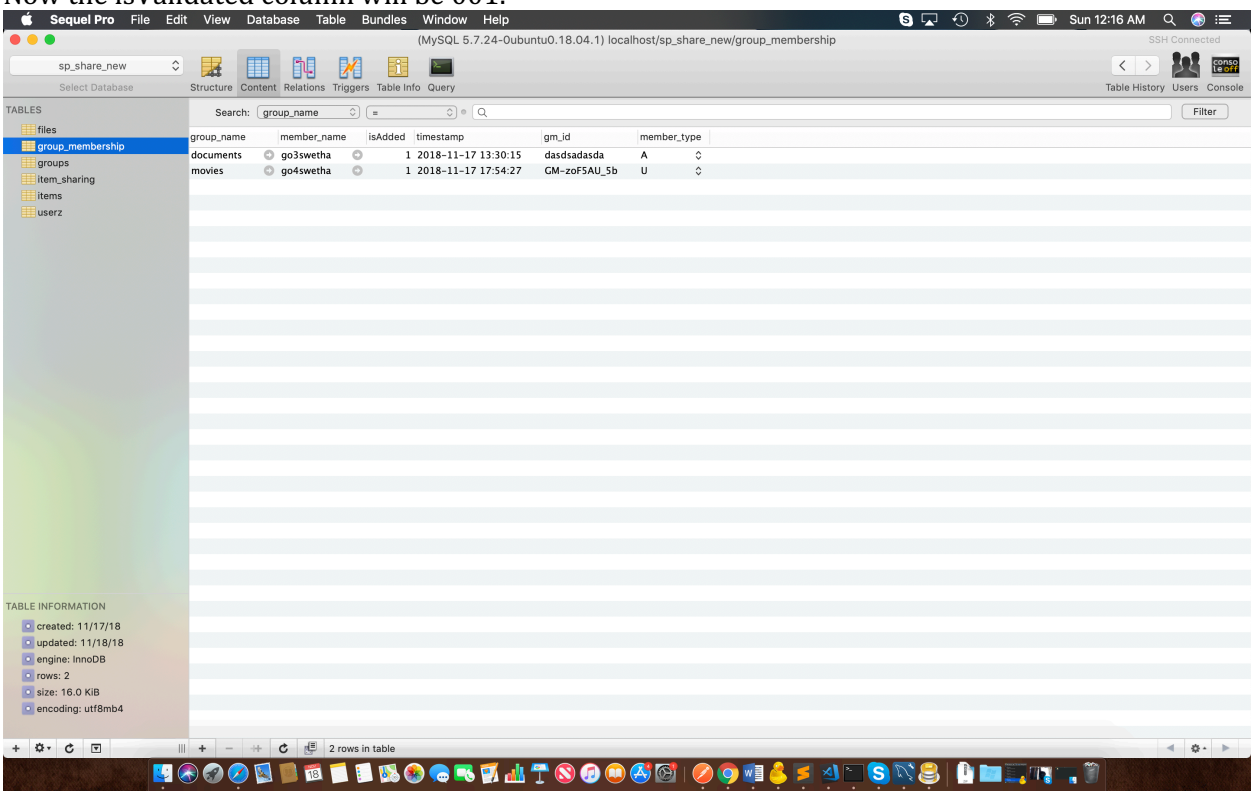


Now the user has to be approved under a group name requested by admin or root. This is done under makeGroupActive

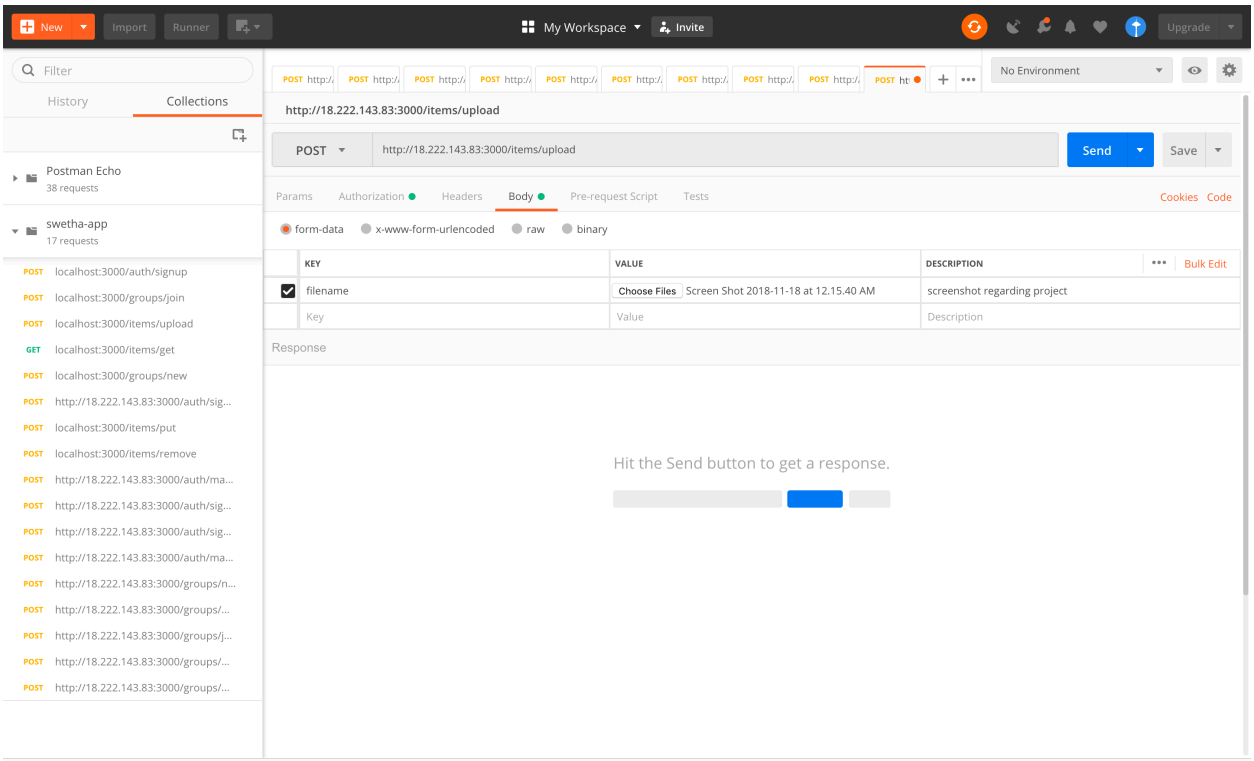


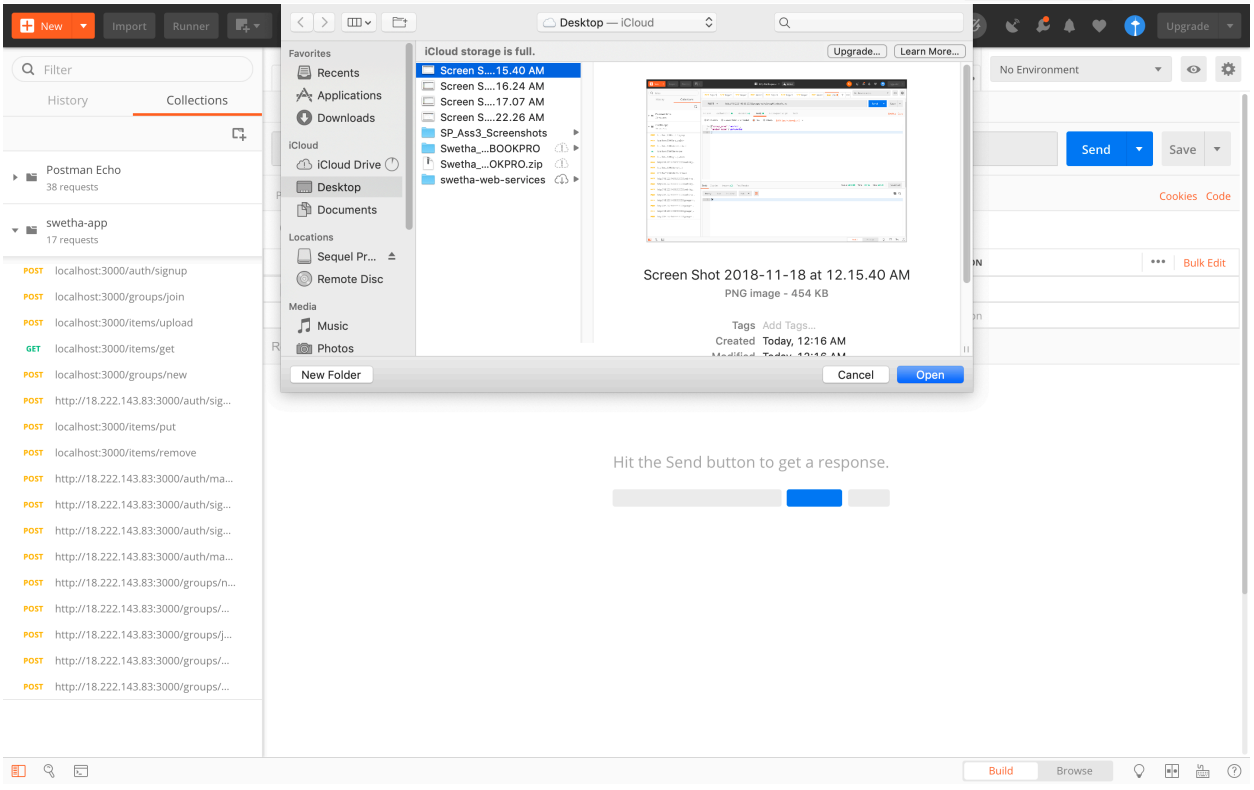


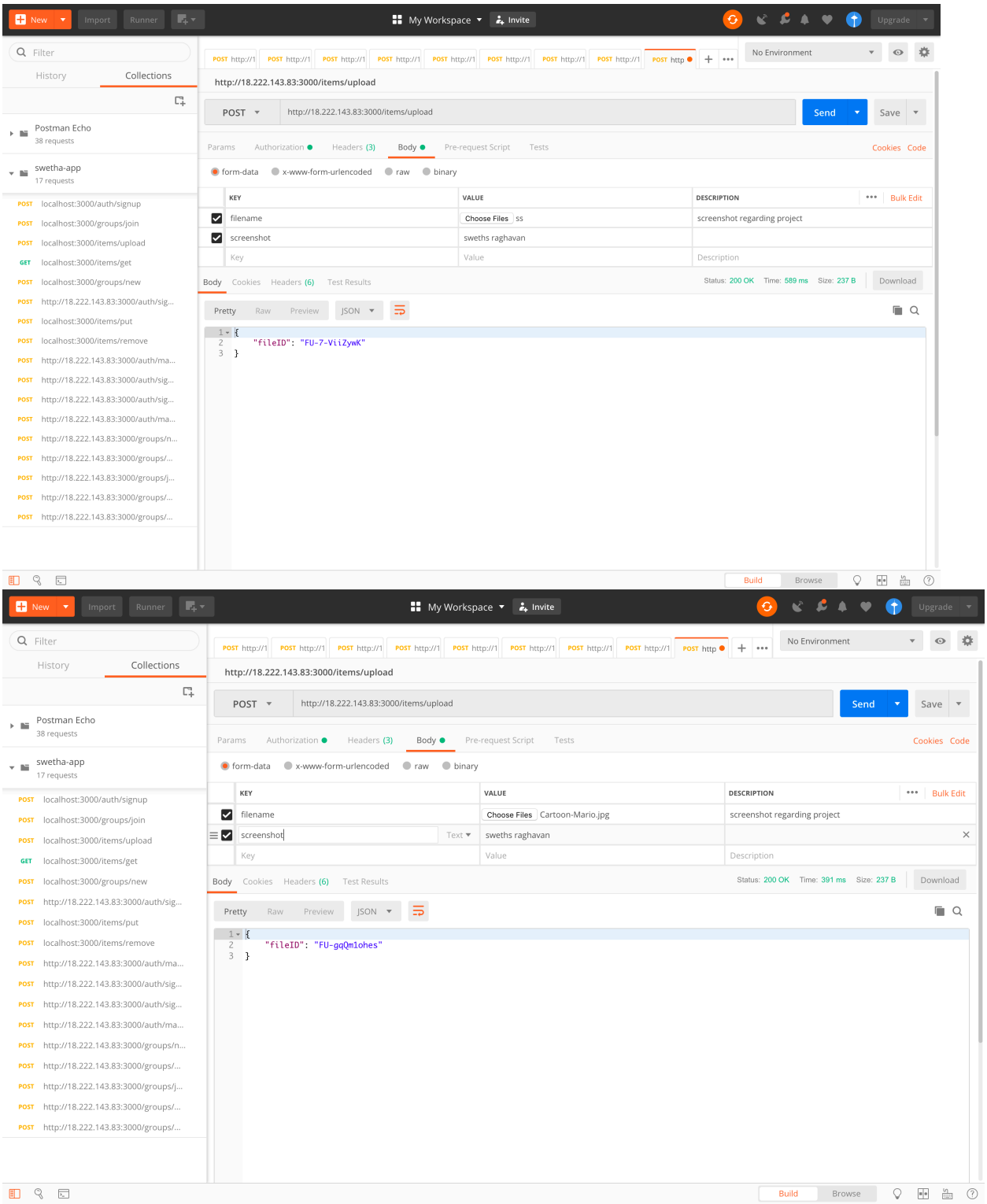
Now the isValidated column will be 001.



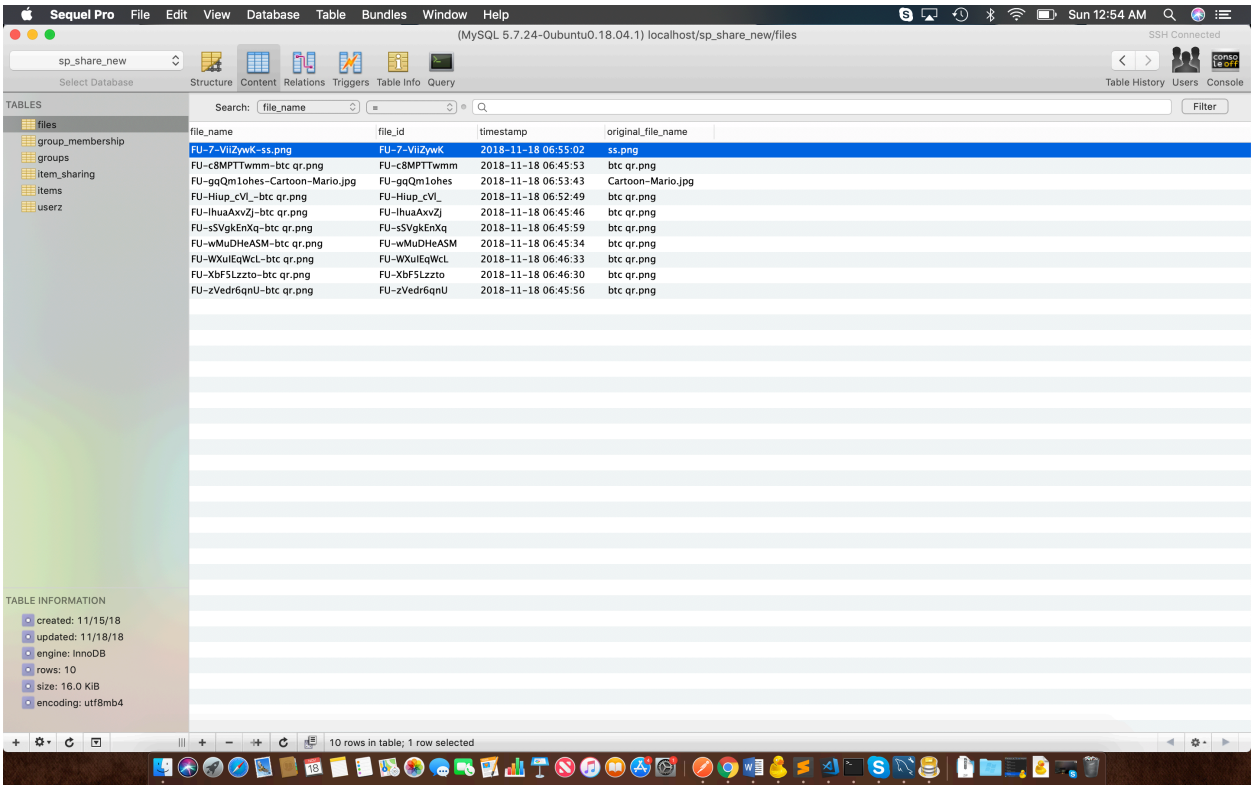
Uploading (File based)

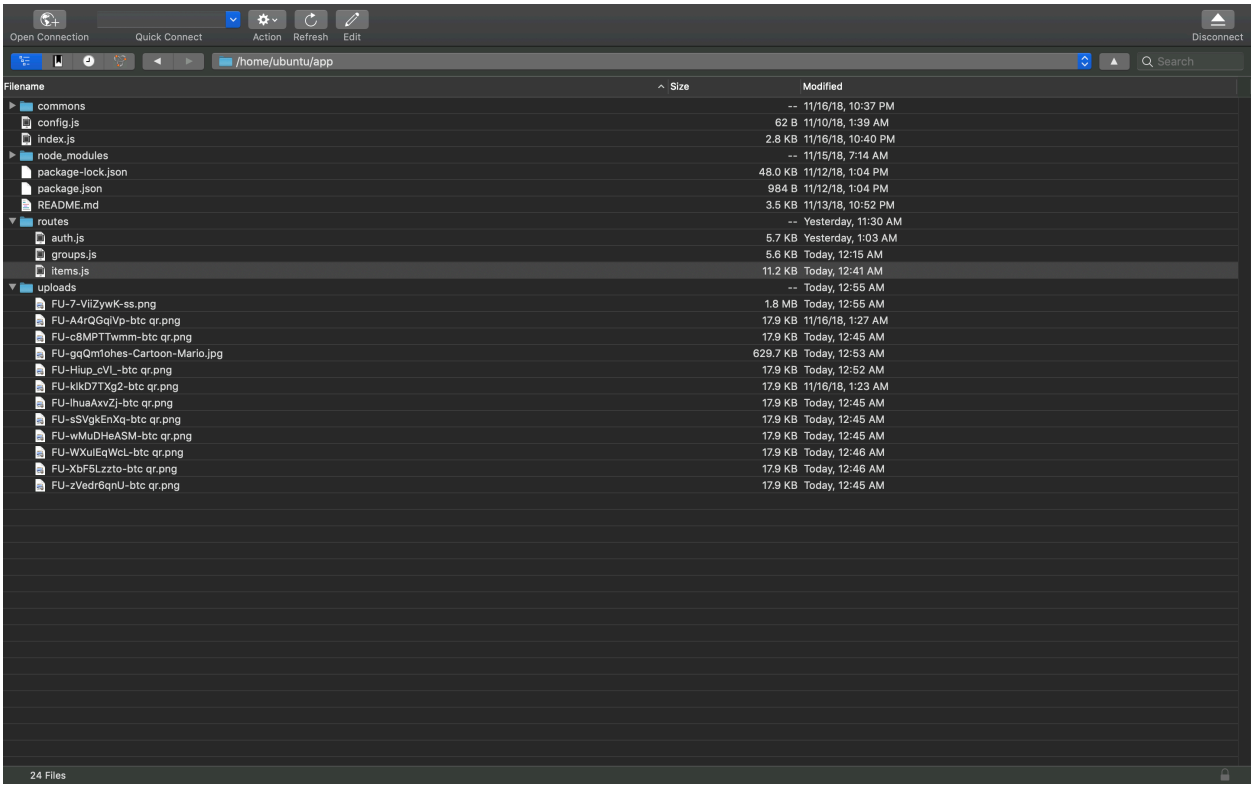

New
Import
Runner
My Workspace
Invite
Upgrade
Filter
History
Collections
Postman Echo
38 requests
swetha-app
17 requests
POST localhost:3000/auth/signup
POST localhost:3000/groups/join
POST localhost:3000/items/upload
GET localhost:3000/items/get
POST localhost:3000/groups/new
POST http://18.222.143.83:3000/auth/sig...
POST localhost:3000/items/put
POST localhost:3000/items/remove
POST http://18.222.143.83:3000/auth/ma...
POST http://18.222.143.83:3000/auth/sig...
POST http://18.222.143.83:3000/auth/sig...
POST http://18.222.143.83:3000/auth/ma...
POST http://18.222.143.83:3000/groups/n...
POST http://18.222.143.83:3000/groups/...
POST http://18.222.143.83:3000/groups/j...
POST http://18.222.143.83:3000/groups/...
POST http://18.222.143.83:3000/groups/...
No Environment
http://18.222.143.83:3000/items/upload
POST
http://18.222.143.83:3000/items/upload
Send
Save
Params
Authorization
Headers
Body
Pre-request Script
Tests
Cookies
Code
form-data
x-www-form-urlencoded
raw
binary
KEY
VALUE
DESCRIPTION
Bulk Edit
filename
Choose Files
Screen Shot 2018-11-18 at 12.15.40 AM
screenshot regarding project
Key
Value
Description
Response
Hit the Send button to get a response.
Build
Browse


New
Import
Runner
Desktop — iCloud
iCloud storage is full.
Upgrade...
Learn More...
Favorites
Recents
Applications
Downloads
iCloud
iCloud Drive
Desktop
Documents
Locations
Sequel Pr...
Remote Disc
Media
Music
Photos
Screen S....15.40 AM
Screen S....16.24 AM
Screen S....17.07 AM
Screen S....22.26 AM
SP_Ass3_Screenshots
Swetha_...BOOKPRO
Swetha_...OKPRO.zip
swetha-web-services
Screen Shot 2018-11-18 at 12.15.40 AM
PNG image - 454 KB
Tags Add Tags...
Created Today, 12:16 AM
New Folder
Cancel
Open
Filter
History
Collections
Postman Echo
38 requests
swetha-app
17 requests
No Environment
Send
Save
Cookies
Code
Bulk Edit
Hit the Send button to get a response.
Build
Browse

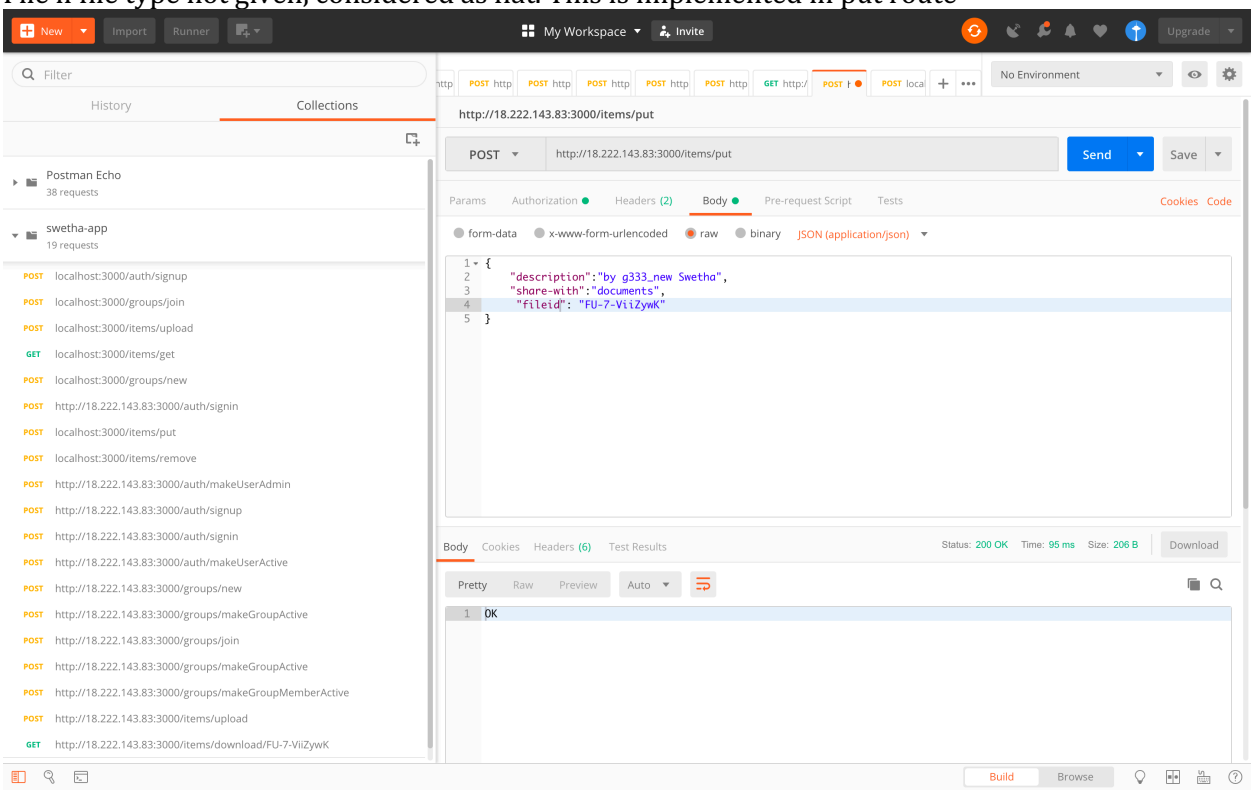


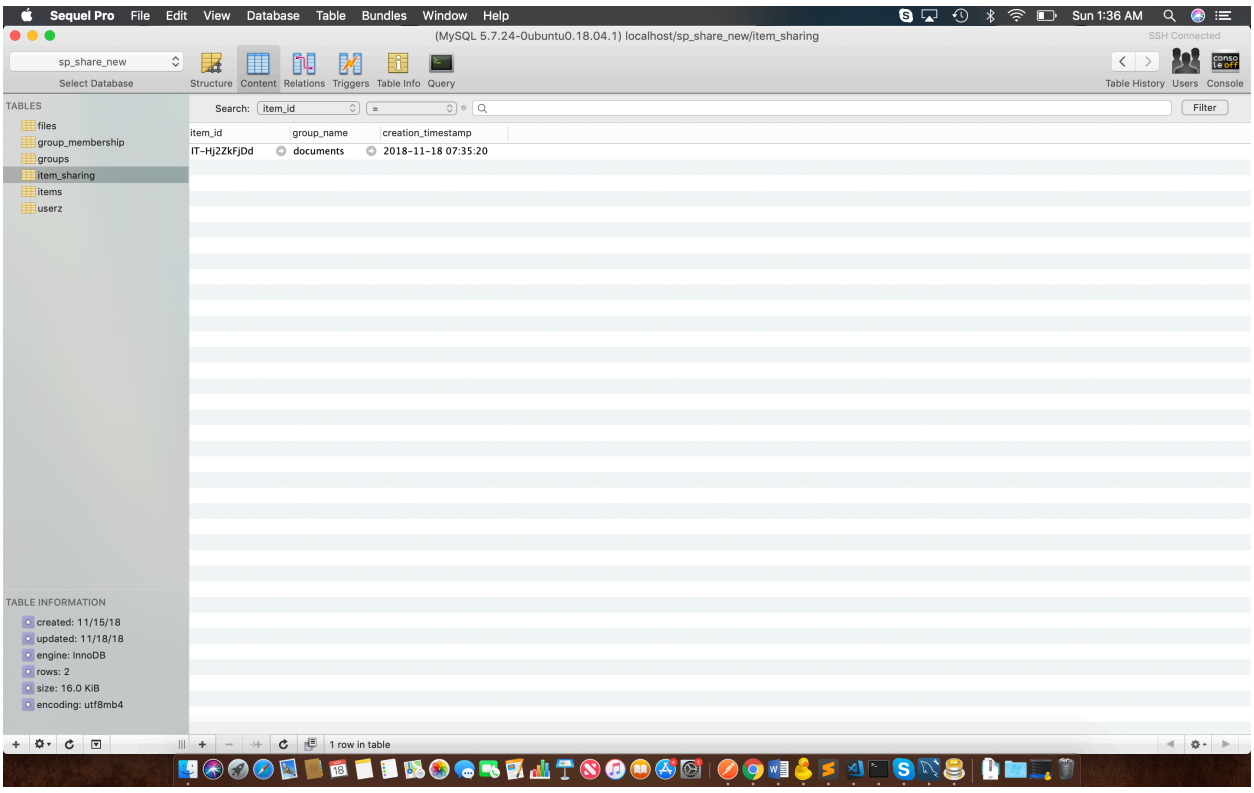
File is uploaded. Along with a short id.


Sequel Pro File Edit View Database Table Bundles Window Help
Sun 12:54 AM
(MySQL 5.7.24-0ubuntu0.18.04.1) localhost/sp_share_new/files
SSH Connected
sp_share_new
Select Database
Structure Content Relations Triggers Table Info Query
Table History Users Console
TABLES
files
group_membership
groups
item_sharing
items
userz
Search: file_name
Filter
file_name file_id timestamp original_file_name
FU-7-ViiZywK-ss.png FU-7-ViiZywK 2018-11-18 06:55:02 ss.png
FU-c8MPTTwmm-btc qr.png FU-c8MPTTwmm 2018-11-18 06:45:53 btc qr.png
FU-gqQm1ohes-Cartoon-Mario.jpg FU-gqQm1ohes 2018-11-18 06:53:43 Cartoon-Mario.jpg
FU-Hiup_cVl_-btc qr.png FU-Hiup_cVl_ 2018-11-18 06:52:49 btc qr.png
FU-lhuaAxvZj-btc qr.png FU-lhuaAxvZj 2018-11-18 06:45:46 btc qr.png
FU-sSVgkEnXq-btc qr.png FU-sSVgkEnXq 2018-11-18 06:45:59 btc qr.png
FU-wMuDHeASM-btc qr.png FU-wMuDHeASM 2018-11-18 06:45:34 btc qr.png
FU-WXulEqWcL-btc qr.png FU-WXulEqWcL 2018-11-18 06:46:33 btc qr.png
FU-XbF5Lzzto-btc qr.png FU-XbF5Lzzto 2018-11-18 06:46:30 btc qr.png
FU-zVedr6qnU-btc qr.png FU-zVedr6qnU 2018-11-18 06:45:56 btc qr.png
TABLE INFORMATION
created: 11/15/18
updated: 11/18/18
engine: InnoDB
rows: 10
size: 16.0 KiB
encoding: utf8mb4
10 rows in table; 1 row selected


Open Connection
Quick Connect
Action
Refresh
Edit
Disconnect
/home/ubuntu/app
Search
Filename
Size
Modified
commons -- 11/16/18, 10:37 PM
config.js 62 B 11/10/18, 1:39 AM
index.js 2.8 KB 11/16/18, 10:40 PM
node_modules -- 11/15/18, 7:14 AM
package-lock.json 48.0 KB 11/12/18, 1:04 PM
package.json 984 B 11/12/18, 1:04 PM
README.md 3.5 KB 11/13/18, 10:52 PM
routes -- Yesterday, 11:30 AM
auth.js 5.7 KB Yesterday, 1:03 AM
groups.js 5.6 KB Today, 12:15 AM
items.js 11.2 KB Today, 12:41 AM
uploads -- Today, 12:55 AM
FU-7-ViiZywK-ss.png 1.8 MB Today, 12:55 AM
FU-A4rQGqiVp-btc qr.png 17.9 KB 11/16/18, 1:27 AM
FU-c8MPTTwmm-btc qr.png 17.9 KB Today, 12:45 AM
FU-gqQm1ohes-Cartoon-Mario.jpg 629.7 KB Today, 12:53 AM
FU-Hiup_cVl_-btc qr.png 17.9 KB Today, 12:52 AM
FU-klkD7TXg2-btc qr.png 17.9 KB 11/16/18, 1:23 AM
FU-lhuaAxvZj-btc qr.png 17.9 KB Today, 12:45 AM
FU-sSVgkEnXq-btc qr.png 17.9 KB Today, 12:45 AM
FU-wMuDHeASM-btc qr.png 17.9 KB Today, 12:45 AM
FU-WXulEqWcL-btc qr.png 17.9 KB Today, 12:46 AM
FU-XbF5Lzzto-btc qr.png 17.9 KB Today, 12:46 AM
FU-zVedr6qnU-btc qr.png 17.9 KB Today, 12:45 AM
24 Files

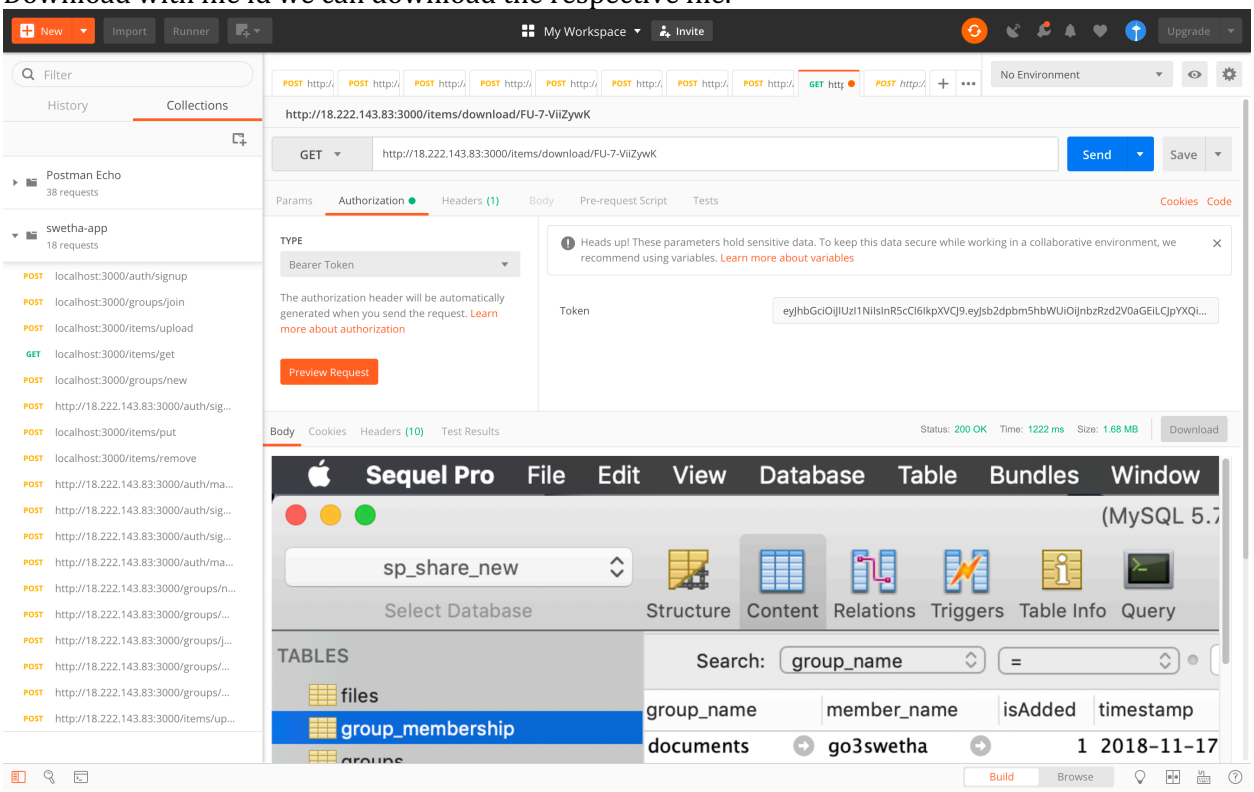
File if file type not given, considered as flat. This is implemented in put route





```
//

// Share Schema
// items: pictures, videos, text, and other format "binary" (raw) files.

router.post('/put', function (req, res) {
    let xx = req.header('authorization');
    let TokenArray = xx.split(" ");
    let token = TokenArray[1];
    let itemBody = req.body
    let item_description = connection.escape(itemBody.description);
    let item_type = itemBody.fileid ? 'FILE' : 'FLAT';
    let file_id = connection.escape(itemBody.fileid);
    let item_id = connection.escape('IT-' + shortid.generate());
    let shared_group = connection.escape(itemBody['share-with']);


    let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
    _bav(function () {
        let loginname = connection.escape(req.local.loginname);
        if (item_type == 'FLAT') {
            item_type = connection.escape(item_type);
            connection.query("INSERT INTO `sp_share_new`.`items` (`item_id`, `item_type`, `item_description`, `item_creator`) VALUES (" + item_id + ", " + item_type + ", " + item_description + "," + loginname + ")", function (error, result) {
                if (!error) {
                    connection.query("select group_name from sp_share_new.groups where groups.group_name = " + shared_group, function (error, result) {
                        if (result.length > 0) {
                            connection.query("INSERT INTO `sp_share_new`.`item_sharing` (`item_id`, `group_name`) VALUES (" + item_id + "," + shared_group + ")", function (error, result) {
                                if (!error) {
                                    res.sendStatus(200);
                                }

                            })
                        }
                    })

                }
            })
        } else {
            item_type = connection.escape(item_type);
            connection.query("INSERT INTO `sp_share_new`.`items` (`item_id`, `item_type`, `file_id`, `item_description`, `item_creator`) VALUES (" + item_id + ", " + item_type + ", " + file_id + "," + item_description + "," + loginname + ")", function (error, result) {
                if (!error) {
                    connection.query("select group_name from sp_share_new.groups where groups.group_name = " + shared_group, function (error, result) {
                        if (result.length > 0) {
                            connection.query("INSERT INTO `sp_share_new`.`item_sharing` (`item_id`, `group_name`) VALUES (" + item_id + "," + shared_group + ")", function (error, result) {
                                if (!error) {
                                    res.sendStatus(200);
                                }

                            })
                        }
                    })

                }
            })
        }
    }
```

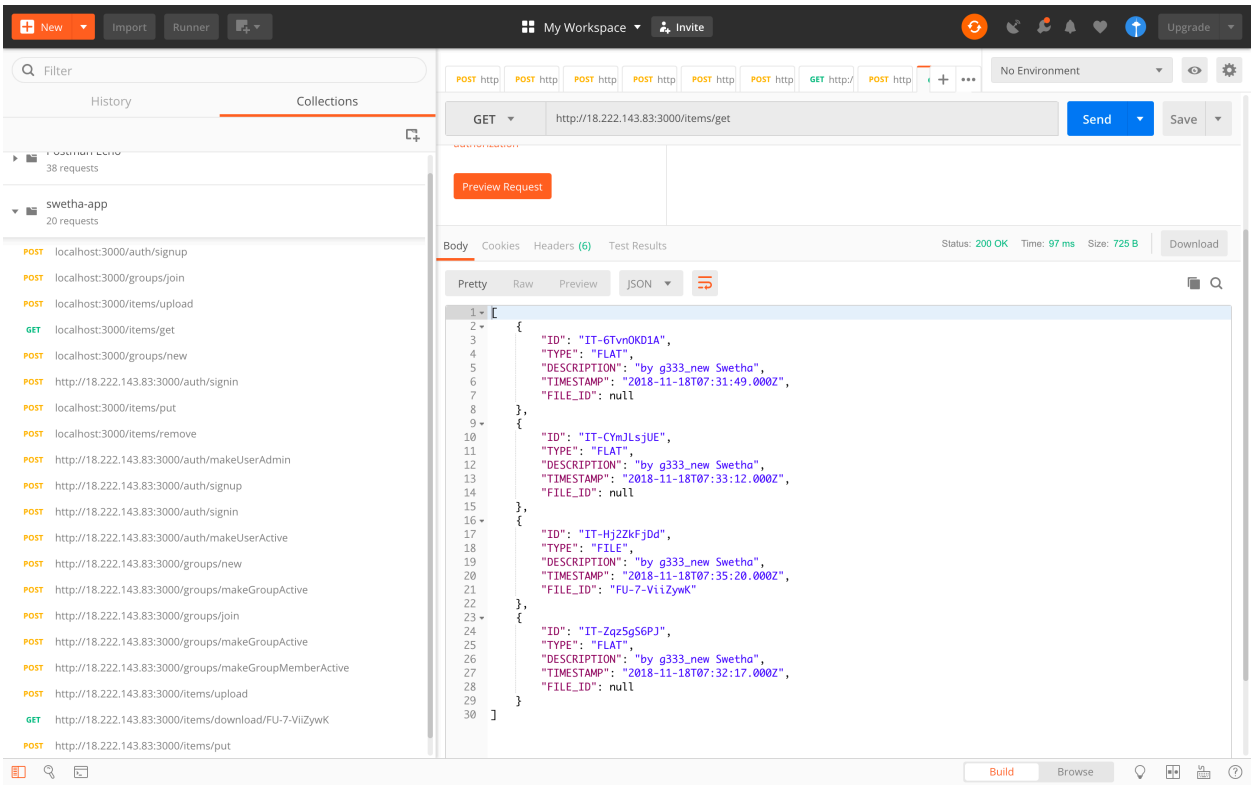
Download with file id we can download the respective file.



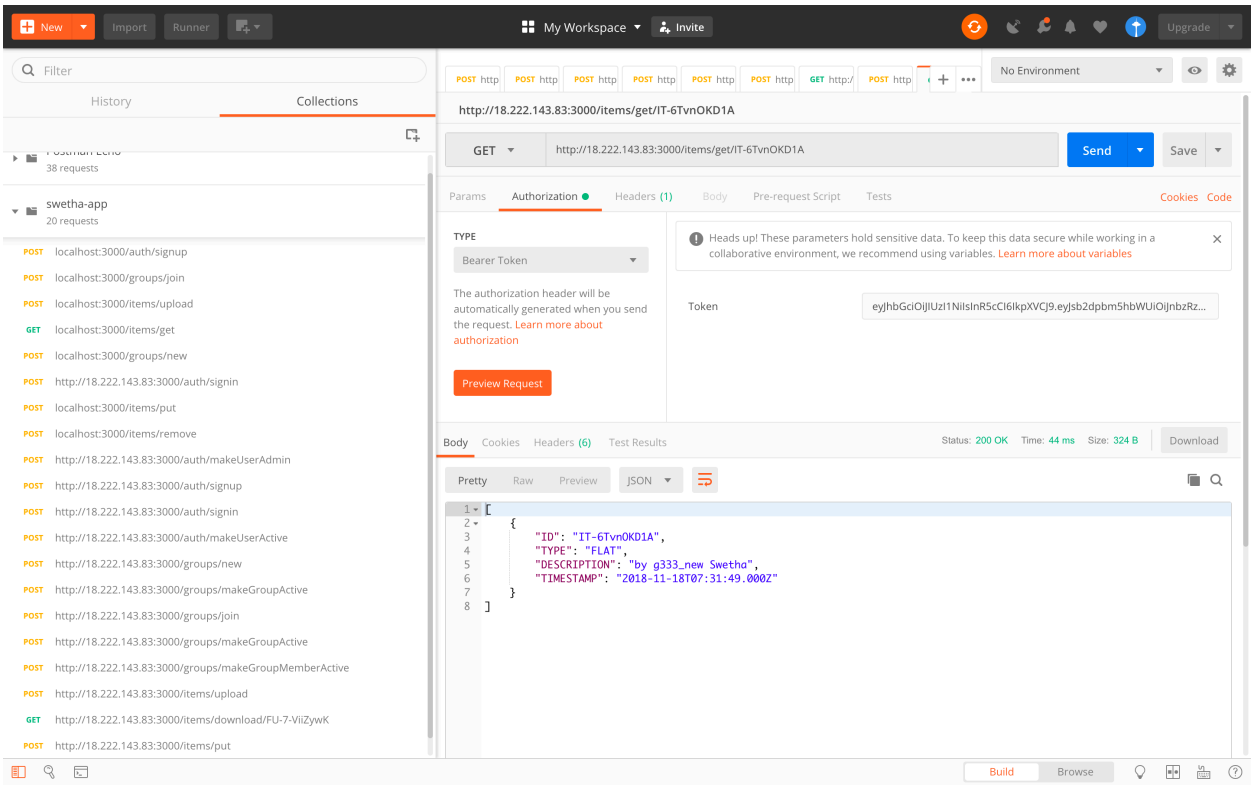
```
router.get('/download/:file_id', function (req, res) {
    let xx = req.header('authorization');
    let TokenArray = xx.split(" ");
    let token = TokenArray[1];
    let file_id = connection.escape(req.params.file_id);
    let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
    _bav(function () {
        let login_name = connection.escape(req.local.loginname);
        connection.query("select group_membership.member_name from group_membership where group_membership.group_name in (select iter
            if (!error) {
                connection.query("select * from files where files.file_id=" + file_id, function (error, result) {
                    let file_path = path.resolve("./uploads/" + result[0].file_name);
                    res.download(file_path);

                })

            }
        })
    })


})
```

Get will extract all meta data of the files along with its file id(if flat-null else fileid), item id, timestamp, item description of all the files the user has uploaded.

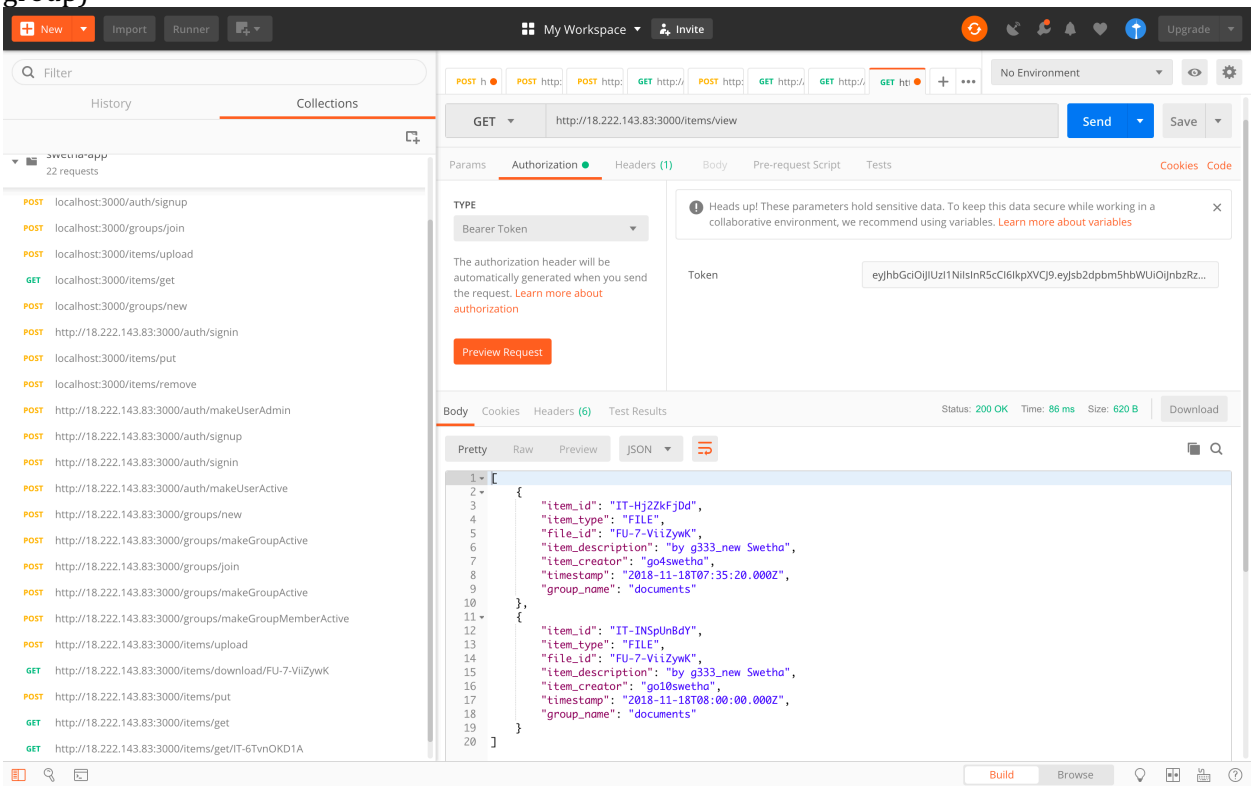
```
router.get('/get', function (req, res) {
        let xx = req.header('authorization');
        let TokenArray = xx.split(" ");
        let token = TokenArray[1];
        let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
        _bav(function (params) {
            let loginname = connection.escape(req.local.loginname);
            connection.query("select item_id AS 'ID',item_type AS
'TYPE',item_description AS 'DESCRIPTION',timestamp AS 'TIMESTAMP',file_id AS 'FILE_ID'
from sp_share_new.items where sp_share_new.items.item_creator = " + loginname,
function (error, result) {
                if (!error) {
                    res.send(JSON.parse(JSON.stringify(result)));
                }
            })
        });
    })
```




```
router.get('/get/:item_id', function (req, res) {
        let xx = req.header('authorization');
        let item_id = connection.escape(req.params.item_id);
        let TokenArray = xx.split(" ");
        let token = TokenArray[1];
        let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
        _bav(function (params) {
            let loginname = connection.escape(req.local.loginname);
            connection.query("select item_id AS 'ID',item_type AS
'TYPE',item_description AS 'DESCRIPTION',timestamp AS 'TIMESTAMP' from
sp_share_new.items where sp_share_new.items.item_creator = " + loginname + " and
sp_share_new.items.item_id =" + item_id, function (error, result) {
                if (!error) {
                    res.send(JSON.parse(JSON.stringify(result)));
                }
            })
        });
    })
```



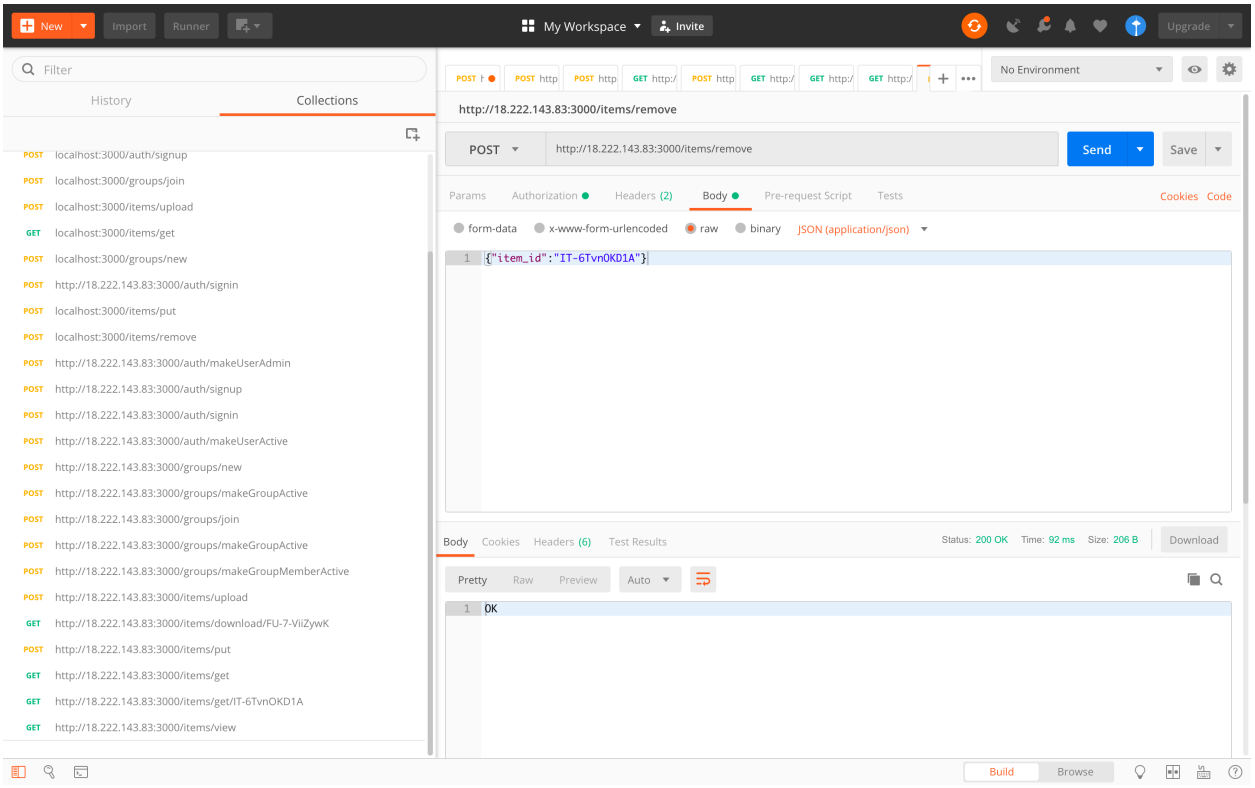
View(if a user belongs to a particular group and this view shows all the files he has shared in that group)

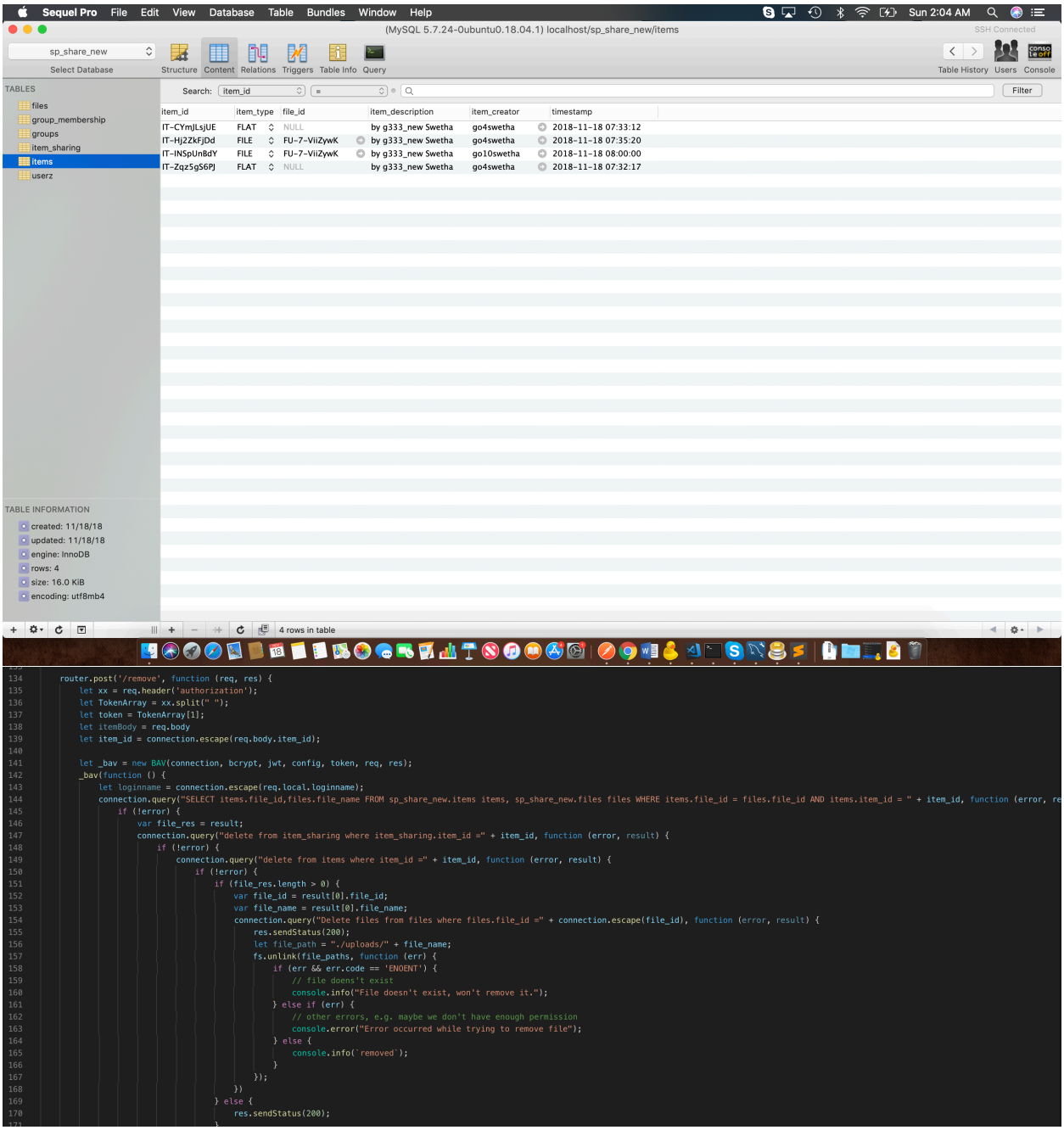



```
router.get('/view', function (req, res) {
        let xx = req.header('authorization');
```

```
        let item_id = connection.escape(req.params.item_id);
        let TokenArray = xx.split(" ");
        let token = TokenArray[1];
        let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
        _bav(function (params) {
            let loginname = connection.escape(req.local.loginname);
            connection.query("select items.*,item_sharing.group_name from
group_membership join item_sharing on item_sharing.group_name =
group_membership.group_name join items on item_sharing.item_id = items.item_id where
group_membership.member_name = " + loginname, function (error, result) {
                if (!error) {
                    res.send(JSON.parse(JSON.stringify(result)));
                }
            })
        });
    })
```

Remove



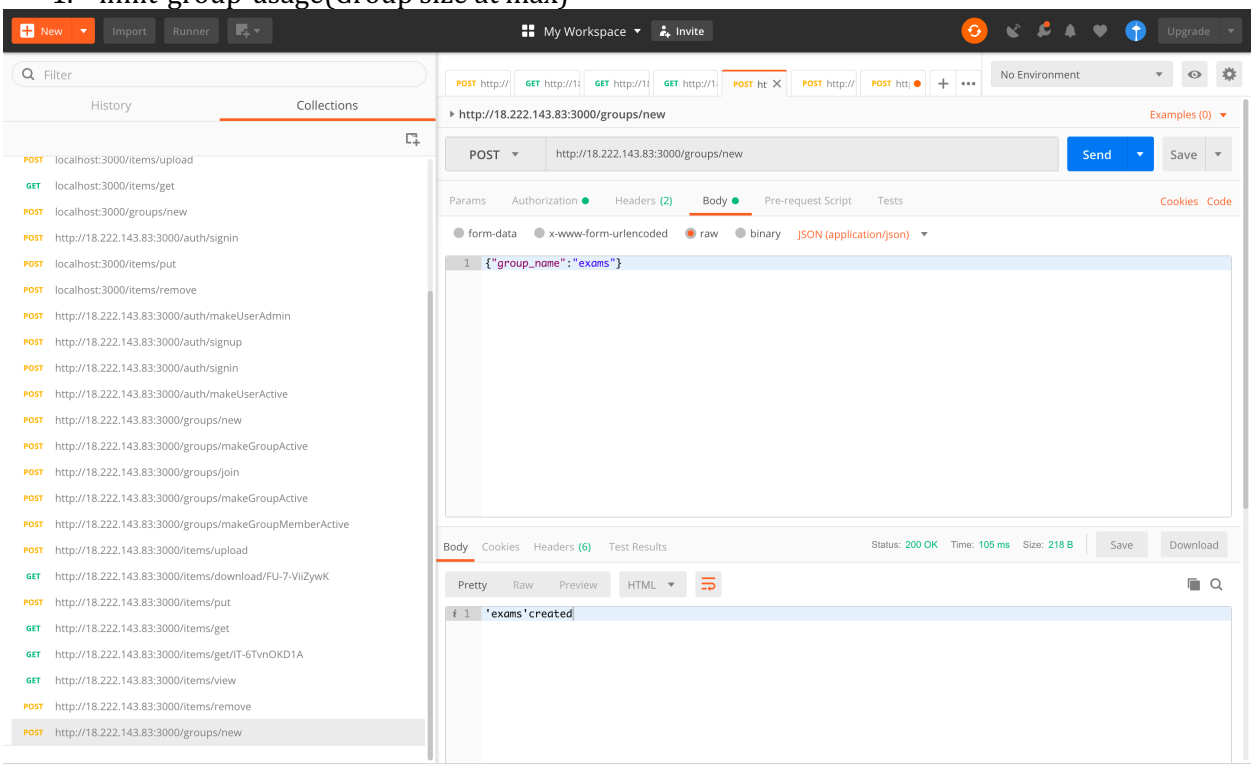


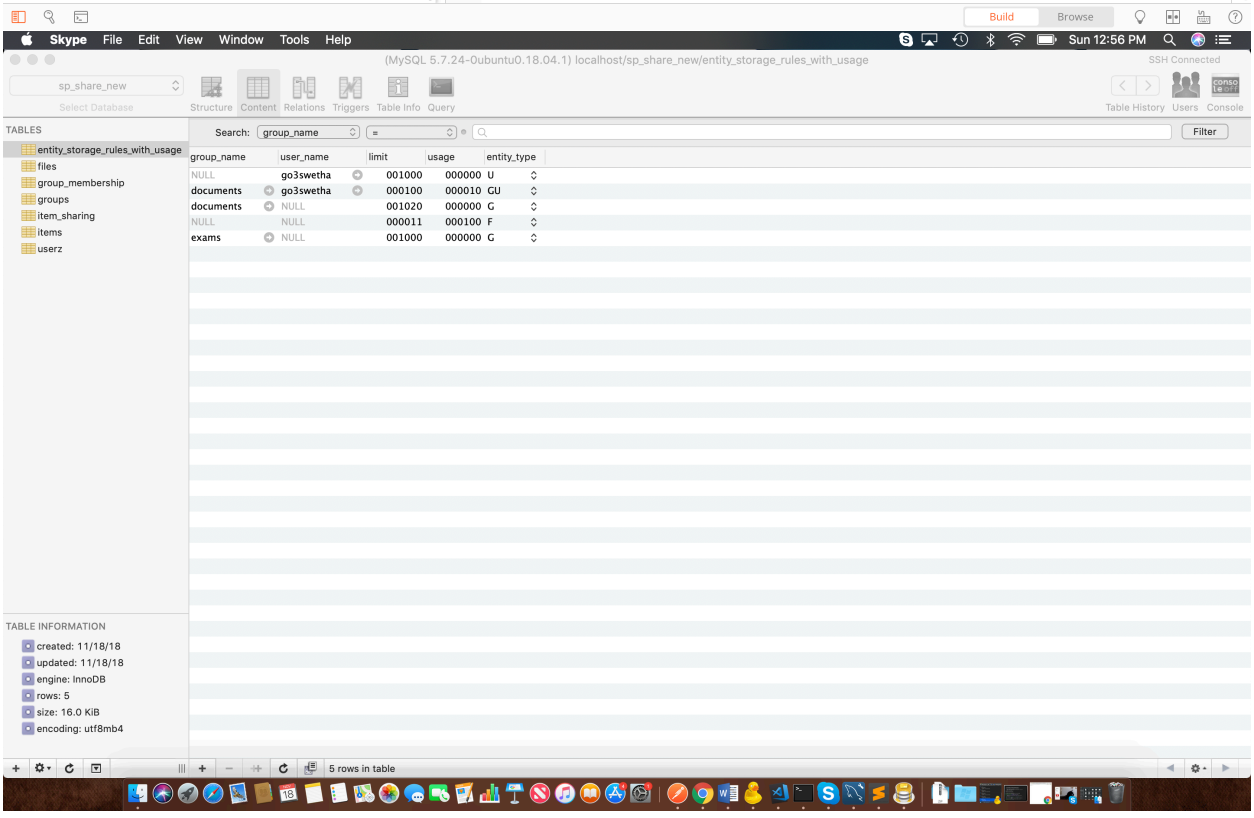

```
router.post('/remove', function (req, res) {
    let xx = req.header('authorization');
    let TokenArray = xx.split(" ");
    let token = TokenArray[1];
    let itemBody = req.body
    let item_id = connection.escape(req.body.item_id);

    let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
    _bav(function () {
        let loginname = connection.escape(req.local.loginname);
        connection.query("SELECT items.file_id,files.file_name FROM sp_share_new.items items, sp_share_new.files files WHERE items.file_id = files.file_id AND items.item_id = " + item_id, function (error, re
            if (!error) {
                var file_res = result;
                connection.query("delete from item_sharing where item_sharing.item_id =" + item_id, function (error, result) {
                    if (!error) {
                        connection.query("delete from items where item_id =" + item_id, function (error, result) {
                            if (!error) {
                                if (file_res.length > 0) {
                                    var file_id = result[0].file_id;
                                    var file_name = result[0].file_name;
                                    connection.query("Delete files from files where files.file_id =" + connection.escape(file_id), function (error, result) {
                                        res.sendStatus(200);
                                        let file_path = "./uploads/" + file_name;
                                        fs.unlink(file_paths, function (err) {
                                            if (err && err.code == 'ENOENT') {
                                                // file doens't exist
                                                console.info("File doesn't exist, won't remove it.");
                                            } else if (err) {
                                                // other errors, e.g. maybe we don't have enough permission
                                                console.error("Error occurred while trying to remove file");
                                            } else {
                                                console.info(`removed`);
                                            }
                                        });
                                    })
                                } else {
                                    res.sendStatus(200);
                                }
```

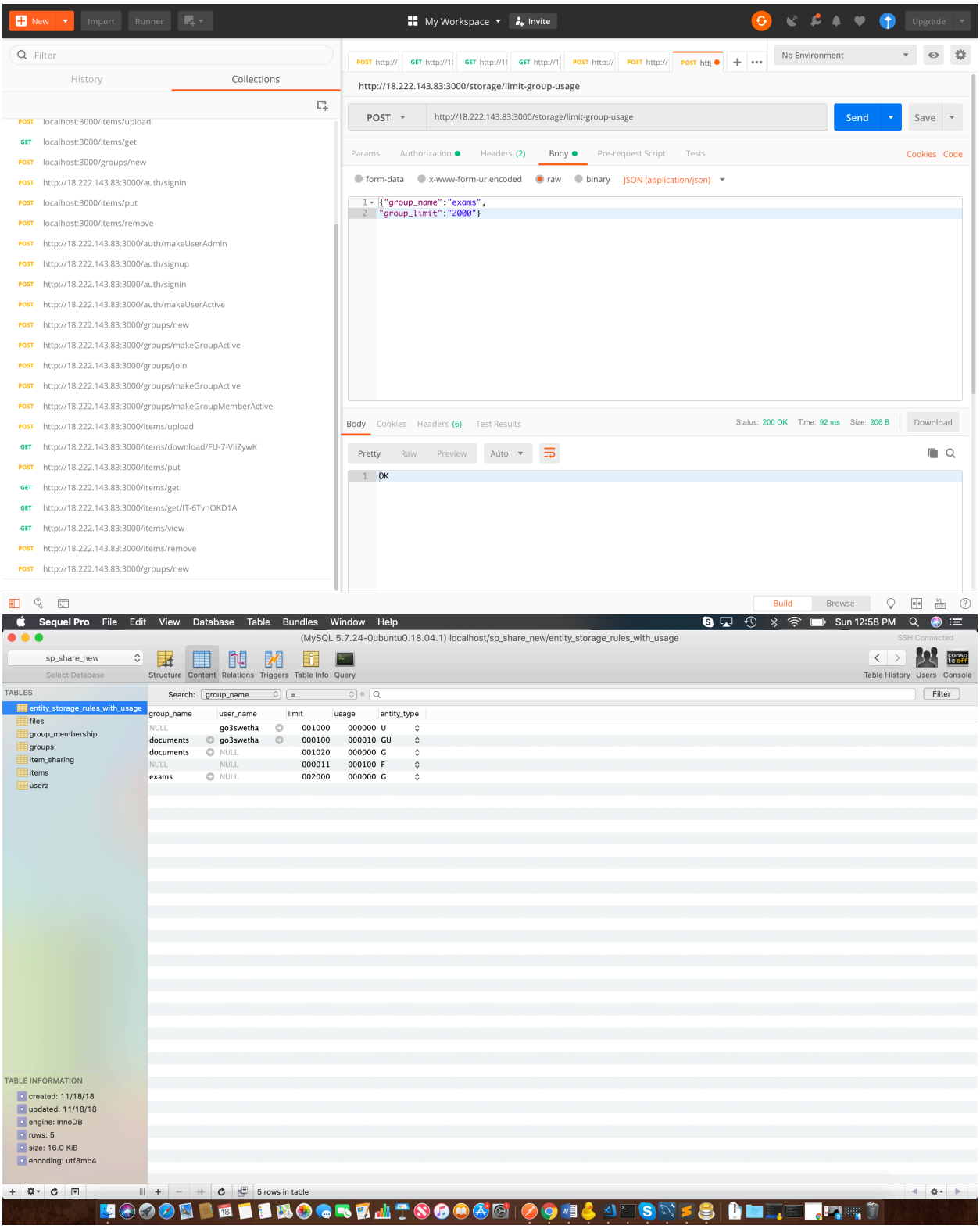
Limit on space and file consumption

1. limit-group-usage(Group size at max)
2. limit-user-usage(User max data)
3. limit-file-usage(File Size max)

1. limit-group-usage(Group size at max)

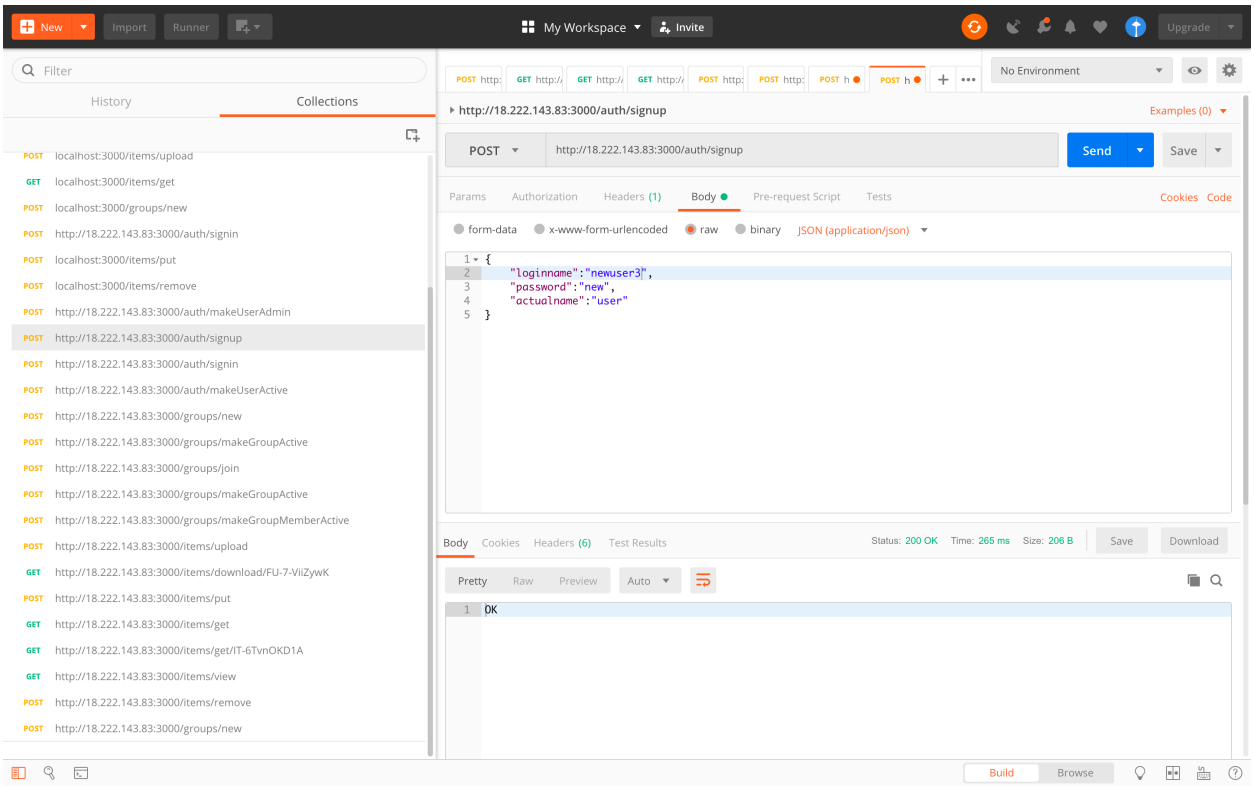


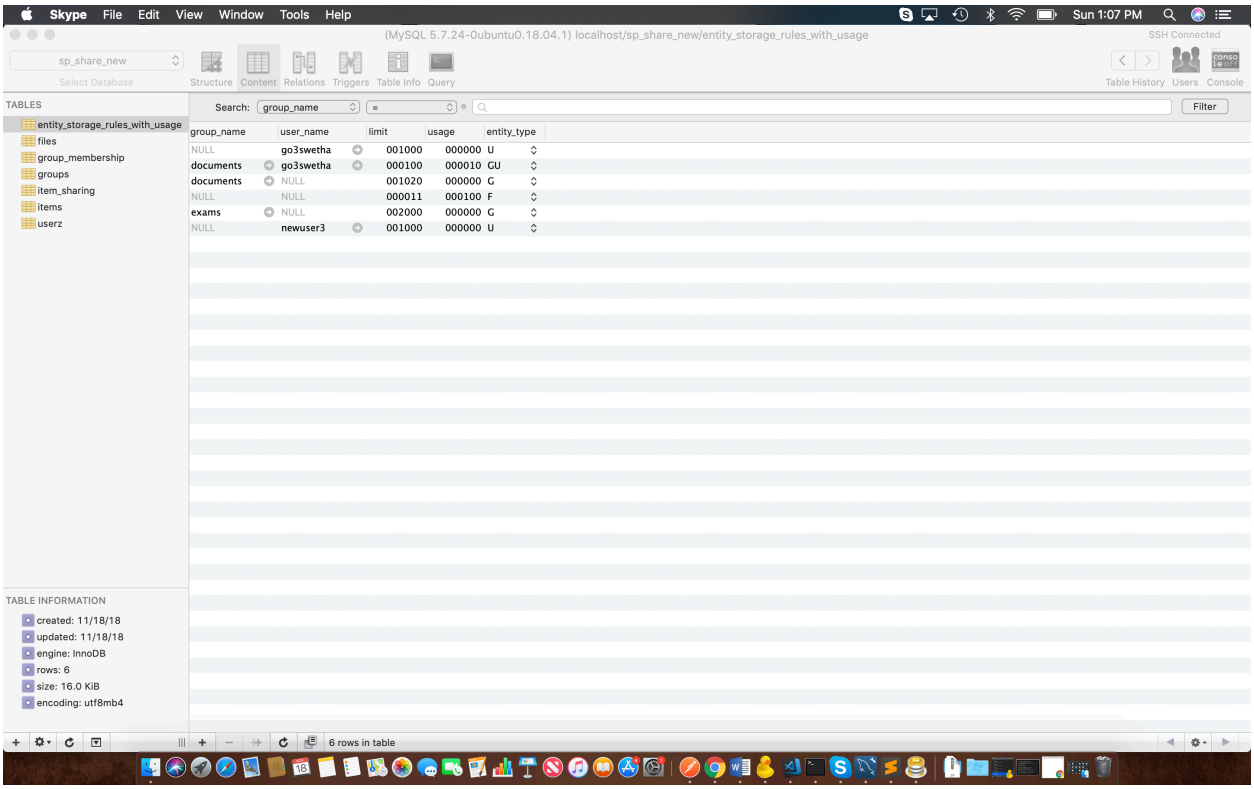


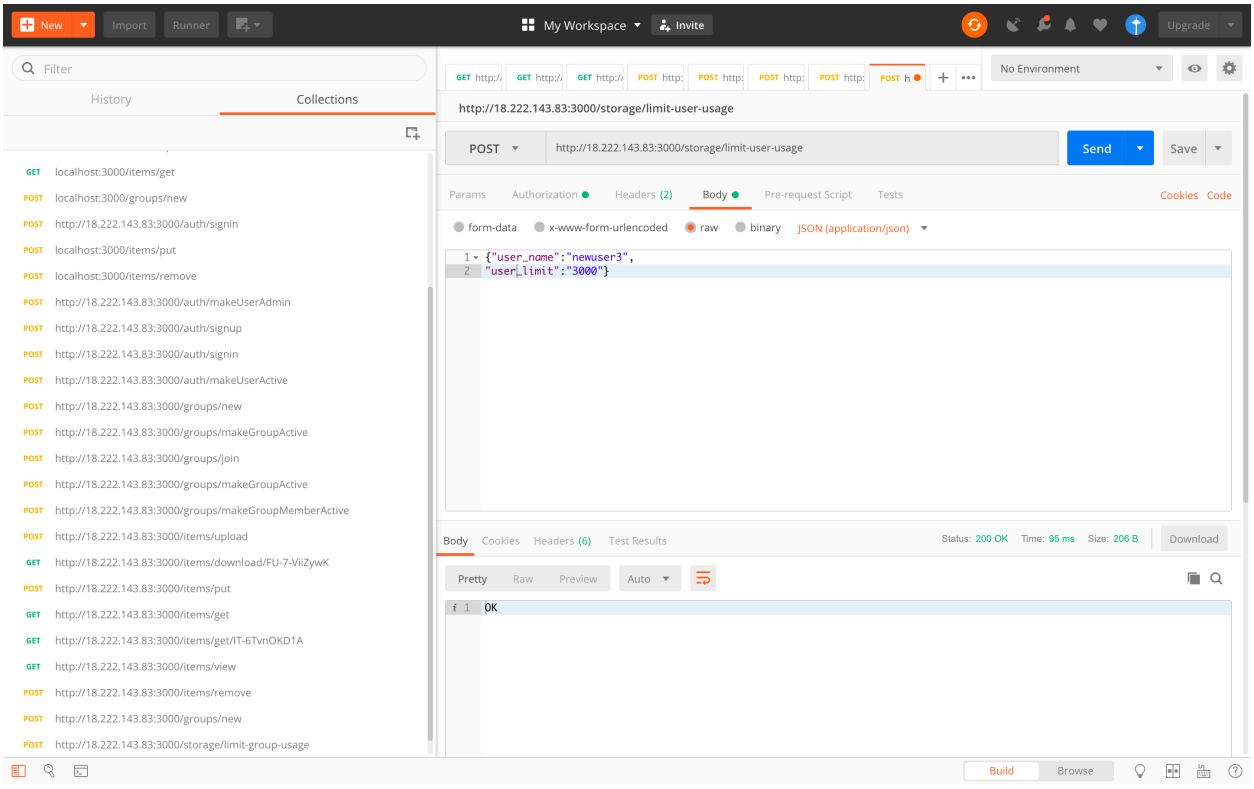

New
Import
Runner
My Workspace
Invite
Upgrade
Filter
History
Collections
POST localhost:3000/items/upload
GET localhost:3000/items/get
POST localhost:3000/groups/new
POST http://18.222.143.83:3000/auth/signin
POST localhost:3000/items/put
POST localhost:3000/items/remove
POST http://18.222.143.83:3000/auth/makeUserAdmin
POST http://18.222.143.83:3000/auth/signup
POST http://18.222.143.83:3000/auth/signin
POST http://18.222.143.83:3000/auth/makeUserActive
POST http://18.222.143.83:3000/groups/new
POST http://18.222.143.83:3000/groups/makeGroupActive
POST http://18.222.143.83:3000/groups/join
POST http://18.222.143.83:3000/groups/makeGroupActive
POST http://18.222.143.83:3000/groups/makeGroupMemberActive
POST http://18.222.143.83:3000/items/upload
GET http://18.222.143.83:3000/items/download/FU-7-ViiZywK
POST http://18.222.143.83:3000/items/put
GET http://18.222.143.83:3000/items/get
GET http://18.222.143.83:3000/items/get/IT-6TvnOKD1A
GET http://18.222.143.83:3000/items/view
POST http://18.222.143.83:3000/items/remove
POST http://18.222.143.83:3000/groups/new
No Environment
http://18.222.143.83:3000/storage/limit-group-usage
POST
http://18.222.143.83:3000/storage/limit-group-usage
Send
Save
Params
Authorization
Headers (2)
Body
Pre-request Script
Tests
Cookies
Code
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
{"group_name":"exams",
"group_limit":"2000"}
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 92 ms
Size: 206 B
Download
Pretty
Raw
Preview
Auto
OK
Build
Browse
Sequel Pro
File
Edit
View
Database
Table
Bundles
Window
Help
Sun 12:58 PM
(MySQL 5.7.24-0ubuntu0.18.04.1) localhost/sp_share_new/entity_storage_rules_with_usage
SSH Connected
sp_share_new
Select Database
Structure
Content
Relations
Triggers
Table Info
Query
Table History
Users
Console
TABLES
entity_storage_rules_with_usage
files
group_membership
groups
item_sharing
items
userz
Search:
group_name
=
Filter
group_name
user_name
limit
usage
entity_type
NULL
go3swetha
001000
000000
U
documents
go3swetha
000100
000010
GU
documents
NULL
001020
000000
G
NULL
NULL
000011
000100
F
exams
NULL
002000
000000
G
TABLE INFORMATION
created: 11/18/18
updated: 11/18/18
engine: InnoDB
rows: 5
size: 16.0 KiB
encoding: utf8mb4
5 rows in table

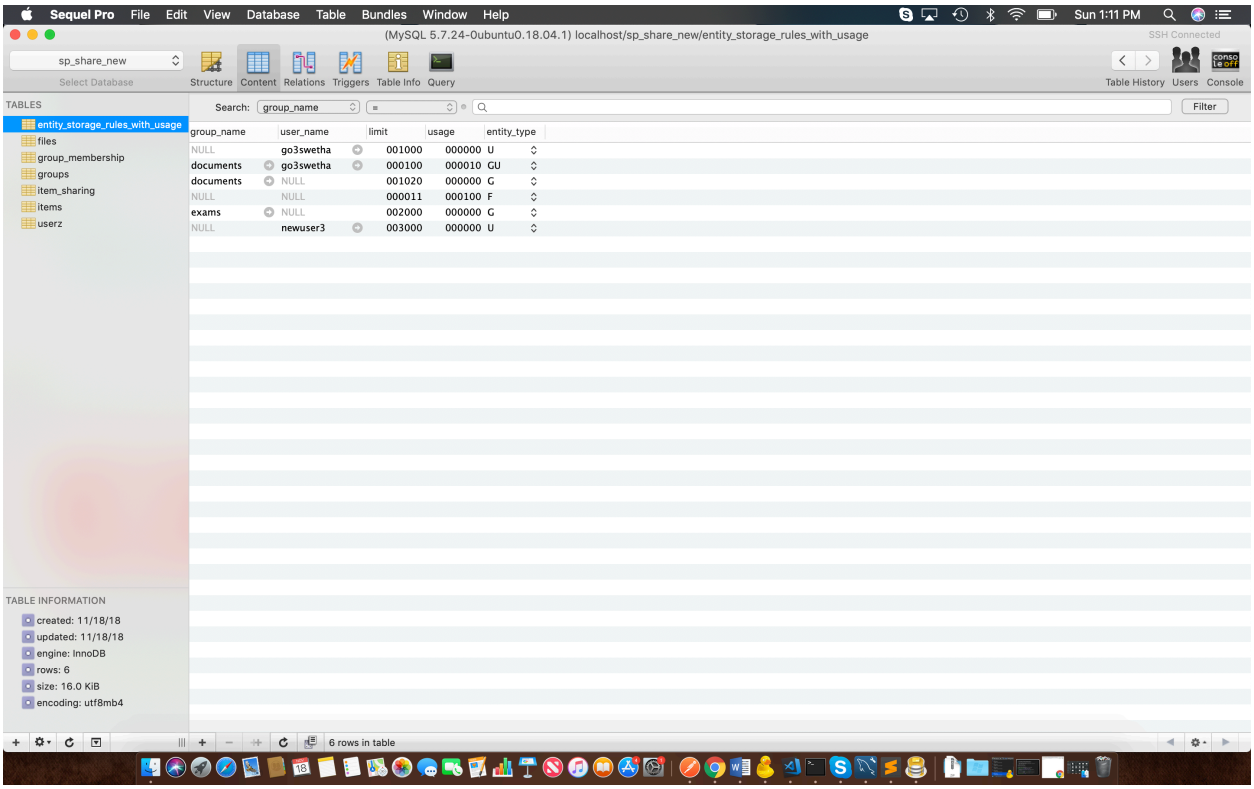
```
module.exports = function (connection, bcrypt, jwt, config, shortid) {
    router.post('/limit-group-usage', function (req, res) {
        let xx = req.header('authorization');
        let TokenArray = xx.split(" ");
        let token = TokenArray[1];
        let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
        let group_name = req.body.group_name;
        let group_limit = req.body.group_limit;
        _bav(function (params) {
            connection.query("UPDATE `sp_share_new`.`entity_storage_rules_with_usage` SET `limit`=" + connection.escape(group_limit) + " WHERE  `group_name`=" + connection.esc
                if (!error) {
                    res.sendStatus(200);
                } else {
                    res.sendStatus(500);
                }
            })

        })
    });
```

2. limit-user-usage(User max data)




Skype File Edit View Window Tools Help
Sun 1:07 PM
(MySQL 5.7.24-0ubuntu0.18.04.1) localhost/sp_share_new/entity_storage_rules_with_usage
SSH Connected
sp_share_new
Select Database
Structure
Content
Relations
Triggers
Table Info
Query
Table History
Users
Console
TABLES
entity_storage_rules_with_usage
files
group_membership
groups
item_sharing
items
userz
Search:
group_name
Filter
group_name
user_name
limit
usage
entity_type
NULL
go3swetha
001000
000000
U
documents
go3swetha
000100
000010
GU
documents
NULL
001020
000000
G
NULL
NULL
000011
000100
F
exams
NULL
002000
000000
G
NULL
newuser3
001000
000000
U
TABLE INFORMATION
created: 11/18/18
updated: 11/18/18
engine: InnoDB
rows: 6
size: 16.0 KiB
encoding: utf8mb4
6 rows in table

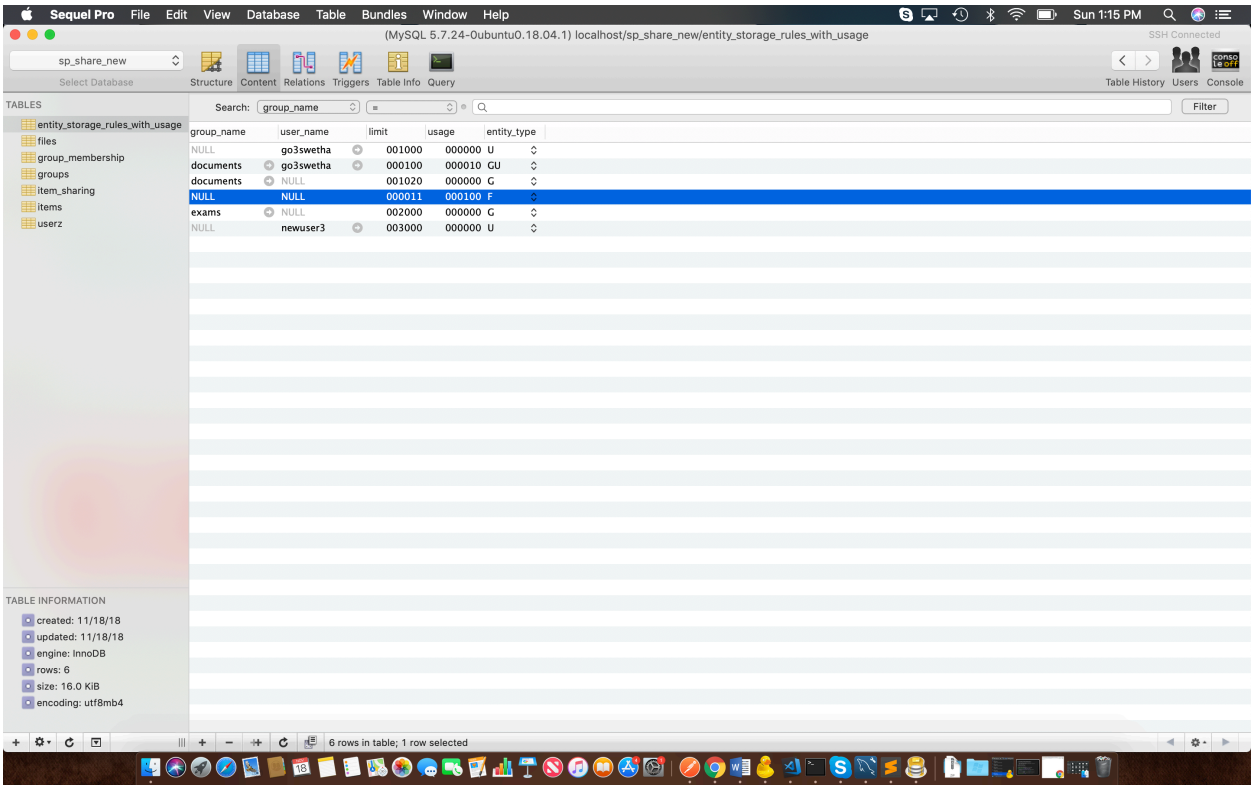

New
Import
Runner
My Workspace
Invite
Upgrade
Filter
History
Collections
GET localhost:3000/items/get
POST localhost:3000/groups/new
POST http://18.222.143.83:3000/auth/signin
POST localhost:3000/items/put
POST localhost:3000/items/remove
POST http://18.222.143.83:3000/auth/makeUserAdmin
POST http://18.222.143.83:3000/auth/signup
POST http://18.222.143.83:3000/auth/signin
POST http://18.222.143.83:3000/auth/makeUserActive
POST http://18.222.143.83:3000/groups/new
POST http://18.222.143.83:3000/groups/makeGroupActive
POST http://18.222.143.83:3000/groups/join
POST http://18.222.143.83:3000/groups/makeGroupActive
POST http://18.222.143.83:3000/groups/makeGroupMemberActive
POST http://18.222.143.83:3000/items/upload
GET http://18.222.143.83:3000/items/download/FU-7-ViiZywK
POST http://18.222.143.83:3000/items/put
GET http://18.222.143.83:3000/items/get
GET http://18.222.143.83:3000/items/get/IT-6TvnOKD1A
GET http://18.222.143.83:3000/items/view
POST http://18.222.143.83:3000/items/remove
POST http://18.222.143.83:3000/groups/new
POST http://18.222.143.83:3000/storage/limit-group-usage
No Environment
http://18.222.143.83:3000/storage/limit-user-usage
POST
http://18.222.143.83:3000/storage/limit-user-usage
Send
Save
Params
Authorization
Headers (2)
Body
Pre-request Script
Tests
Cookies
Code
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
{"user_name":"newuser3",
"user_limit":"3000"}
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 95 ms
Size: 206 B
Download
Pretty
Raw
Preview
Auto
OK
Build
Browse

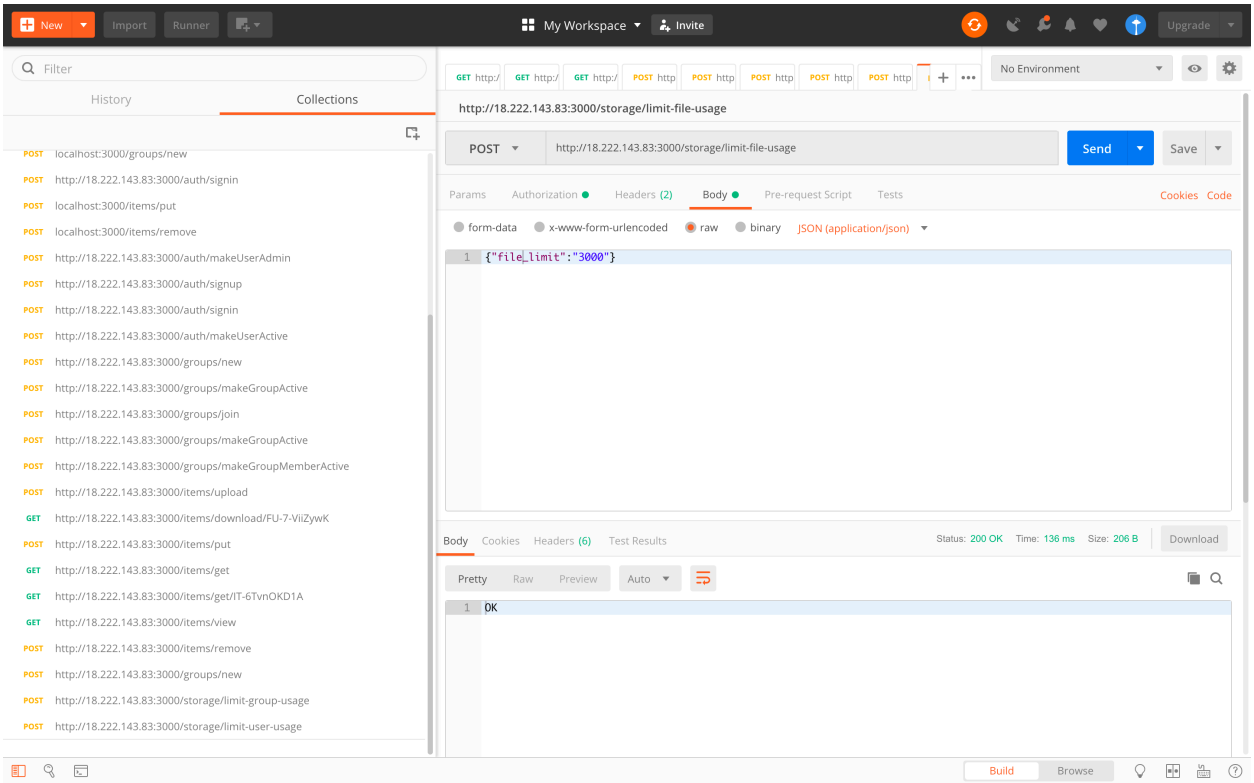



```
    router.post('/limit-user-usage', function (req, res) {
        let xx = req.header('authorization');
        let TokenArray = xx.split(" ");
        let token = TokenArray[1];
        let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
        let user_name = req.body.user_name;
        let user_limit = req.body.user_limit;
        _bav(function (params) {
            connection.query("UPDATE `sp_share_new`.`entity_storage_rules_with_usage` SET `limit`=" + connection.escape(user_limit) + " WHERE  `user_name`=" + connection.escap
                if (!error) {
                    res.sendStatus(200);
                } else {
                    res.sendStatus(500);
                }
            })

        })
    });
```

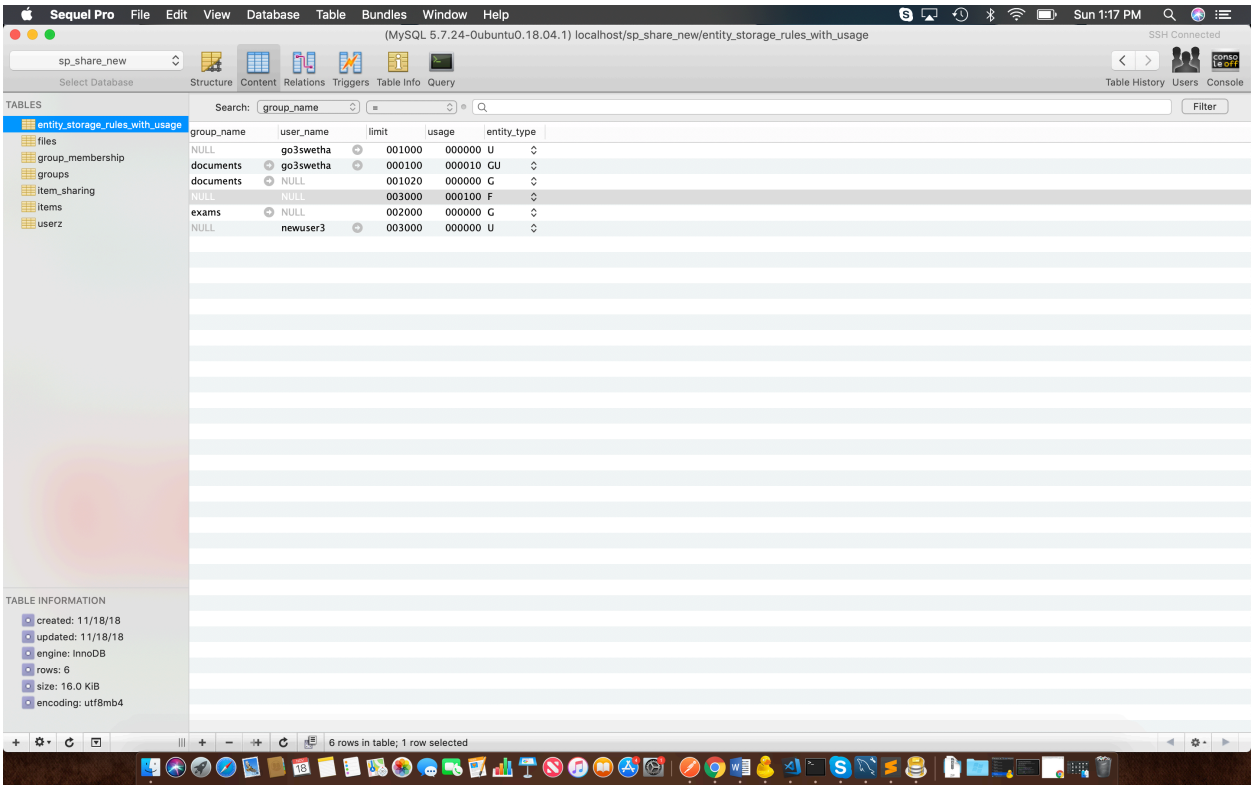
3 limit-file-usage(File Size max)

```
    router.post('/limit-file-usage', function (req, res) {
        let xx = req.header('authorization');
        let TokenArray = xx.split(" ");
        let token = TokenArray[1];
        let _bav = new BAV(connection, bcrypt, jwt, config, token, req, res);
        let file_limit = req.body.file_limit;
        _bav(function (params) {
            connection.query("UPDATE `sp_share_new`.`entity_storage_rules_with_usage` SET `limit`=" + connection.escape(file_limit) + " WHERE  `group_name` IS NULL AND `user_n
                if (!error) {
                    res.sendStatus(200);
                } else {
                    res.sendStatus(500);
                }
            })

        })
    });

    return router;
}
```


Sequel Pro File Edit View Database Table Bundles Window Help
Sun 1:15 PM
(MySQL 5.7.24-0ubuntu0.18.04.1) localhost/sp_share_new/entity_storage_rules_with_usage
SSH Connected
sp_share_new
Select Database
Structure
Content
Relations
Triggers
Table Info
Query
Table History
Users
Console
TABLES
entity_storage_rules_with_usage
files
group_membership
groups
item_sharing
items
userz
Search:
group_name
Filter
group_name
user_name
limit
usage
entity_type
NULL go3swetha 001000 000000 U
documents go3swetha 000100 000010 GU
documents NULL 001020 000000 G
NULL NULL 000011 000100 F
exams NULL 002000 000000 G
NULL newuser3 003000 000000 U
TABLE INFORMATION
created: 11/18/18
updated: 11/18/18
engine: InnoDB
rows: 6
size: 16.0 KiB
encoding: utf8mb4
6 rows in table; 1 row selected


New
Import
Runner
My Workspace
Invite
Upgrade
Filter
History
Collections
No Environment
http://18.222.143.83:3000/storage/limit-file-usage
POST
http://18.222.143.83:3000/storage/limit-file-usage
Send
Save
Params
Authorization
Headers (2)
Body
Pre-request Script
Tests
Cookies
Code
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
{"file_limit":"3000"}
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 136 ms
Size: 206 B
Download
Pretty
Raw
Preview
Auto
OK
Build
Browse
POST localhost:3000/groups/new
POST http://18.222.143.83:3000/auth/signin
POST localhost:3000/items/put
POST localhost:3000/items/remove
POST http://18.222.143.83:3000/auth/makeUserAdmin
POST http://18.222.143.83:3000/auth/signup
POST http://18.222.143.83:3000/auth/signin
POST http://18.222.143.83:3000/auth/makeUserActive
POST http://18.222.143.83:3000/groups/new
POST http://18.222.143.83:3000/groups/makeGroupActive
POST http://18.222.143.83:3000/groups/join
POST http://18.222.143.83:3000/groups/makeGroupActive
POST http://18.222.143.83:3000/groups/makeGroupMemberActive
POST http://18.222.143.83:3000/items/upload
GET http://18.222.143.83:3000/items/download/FU-7-ViiZywK
POST http://18.222.143.83:3000/items/put
GET http://18.222.143.83:3000/items/get
GET http://18.222.143.83:3000/items/get/IT-6TvnOKD1A
GET http://18.222.143.83:3000/items/view
POST http://18.222.143.83:3000/items/remove
POST http://18.222.143.83:3000/groups/new
POST http://18.222.143.83:3000/storage/limit-group-usage
POST http://18.222.143.83:3000/storage/limit-user-usage

After


Sequel Pro File Edit View Database Table Bundles Window Help
Sun 1:17 PM
(MySQL 5.7.24-0ubuntu0.18.04.1) localhost/sp_share_new/entity_storage_rules_with_usage
SSH Connected
sp_share_new
Select Database
Structure
Content
Relations
Triggers
Table Info
Query
Table History
Users
Console
TABLES
entity_storage_rules_with_usage
files
group_membership
groups
item_sharing
items
userz
Search:
group_name
Filter
group_name
user_name
limit
usage
entity_type
NULL
go3swetha
001000
000000
U
documents
go3swetha
000100
000010
GU
documents
NULL
001020
000000
G
NULL
NULL
003000
000100
F
exams
NULL
002000
000000
G
NULL
newuser3
003000
000000
U
TABLE INFORMATION
created: 11/18/18
updated: 11/18/18
engine: InnoDB
rows: 6
size: 16.0 KiB
encoding: utf8mb4
6 rows in table; 1 row selected