

 Home(/)

0x04. Files manager

 Back-end My Planning

JavaScript

ES6

NoSQL

MongoDB

Redis

NodeJS

ExpressJS

Kue



Projects(/projects/current)

 Weight: 1

Project to be done in teams of 2 people (your team: Michael Lyke, Cyril Ugwuoke)

QA Reviews I can make (corrections to review)



Project over - took place from Aug 29, 2024 6:00 AM to Sep 5, 2024 6:00 AM



Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)

☒ Manual QA review was done on Sep 14, 2024 9:15 AM☒ An auto review will be launched at the deadline

Curriculums(/dashboards/my_curriculums)



In a nutshell...

Concepts(/concepts)

- **Contribution:** 100.0%
- **Manual QA review:** 0.0/12 mandatory & 0.0/20 optional
- **Auto QA review:** 35.0/36 mandatory (video_rooms)
- **Altogether: 35.71%**



Servers(/servers)

- Mandatory: 35.71%
- Optional: 100.0%
- Contribution: 100.0%
- Calculation: $100.0\% * (35.71\% + (35.71\% * 0.0\%)) == 35.71\%$



Sandboxes(/user_containers/current)

Overall comment:

Project was failed automatically.



Tools(/dashboards/my_tools)



Video on demand(/dashboards/videos)

This project is a summary of this back-end trimester: authentication, NodeJS, MongoDB, Redis, pagination and background processing.

The objective is to build a simple platform to upload and view files:



- Users(/users/users)
- Use authentication via a token
- List all files



- Upload a new file
- Discord(/https://discord.com/app)
- Change permission of a file
- View a file
- Generate thumbnails for images

You will be guided step by step for building it, but you have some freedoms of implementation, split more files etc... (utils folder will be your friend)

My Profile(/users/my_profile)



Of course, this kind of service already exists in the real life - it's a learning purpose to assemble each piece and build a full product.

Enjoy!

Resources

Read or watch:



- My Planning(/planning/me)
- Node JS getting started (/rltoken/buFPHJYnZjtOrTd610j6Og)
- Process API doc (/rltoken/uYPplj2cPK8pcPOLtV6RuA)



- Express getting started (/rltoken/SujfeWKCWmUMomfETjETeg)
- Projects(/projects/current)
- Mocha documentation (/rltoken/FzEwplmoZiyGvkgKIIZNJw)
- Nodemon documentation (/rltoken/pdNNTX0OLugbhxvP3sLgOw)



- MongoDB (/rltoken/q1x7y_3GskzVAJBtXcSimA)
- QA Reviews I can make (/corrections/to_review)
- Bull (/rltoken/NkHBpGrxnd0sK_fDPMbihg)



- Image thumbnail (/rltoken/KX6cck2nyLpQOTDMLcwxLg)
- Mini Typescript quizzes (/rltoken/0B9Kc-4HDKLue88ShhQIQ)
- Mini Typescript quizzes (/dashboards/my_content_evaluation_quizzes)
- Redis (/rltoken/nqwKRszO8Tkj_ZWW1EFwGw)

Learning Objectives



Curriculums(/dashboards/my_curriculums)

At the end of this project, you are expected to be able to explain to anyone (/rltoken/88vbnogJmkEoxqu-6wAXEw), **without the help of Google**:



Concepts(/concepts)

- how to create an API with Express
- how to authenticate a user



how to store data in MongoDB

- Conference rooms(/dashboards/video_rooms)
- how to store temporary data in Redis
- how to setup and use a background worker



Servers(/servers)

Requirements



Saved boards(/user_containers/current)

- All your code should be written in Visual Studio Code
- All your files will be interpreted/compiled on Ubuntu 18.04 LTS using node (version 12.x.x)



All your files should end with a new line

- Tools(/dashboards/my_tools)
- A README.md file, at the root of the folder of the project, is mandatory
- Your code should use the js extension



Your code will be verified against lint using ESLint

Video on demand(/dashboards/videos)

Provided files



Peers(/users/peers)

package.json

Click to show/hide file contents



Discord(https://discord.com/app)

.eslintrc.js

Click to show/hide file contents



My Profile(/users/my_profile)

babel.config.js



Click to show/hide file contents



Home(/)

Don't forget to run `$ npm install` when you have the `package.json`



My Planning(/planning/me)



Projects(/projects/current)



0. Redis utils

Quizzes I can make(/corrections/to_review)

mandatory



Score: 100.0% (Checks completed: 100.0%)

Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)

Inside the folder `utils`, create a file `redis.js` that contains the class `RedisClient`.

`RedisClient` should have:



Curriculums(/dashboards/my_curriculums)

- the constructor that creates a client to Redis:
 - any error of the redis client must be displayed in the console (you should use `on('error')`)



Concepts(/concepts)

- a function `isAlive` that returns `true` when the connection to Redis is a success otherwise, `false`



Conference rooms(/dashboards/video_rooms)

- an asynchronous function `get` that takes a string key as argument and returns the Redis value stored for this key



Servers(/servers)

- an asynchronous function `set` that takes a string key, a value and a duration in second as arguments to store it in Redis (with an expiration set by the duration argument)

- an asynchronous function `del` that takes a string key as argument and remove the value in Redis



Storage boxes(/user_containers/current)

After the class definition, create and export an instance of `RedisClient` called `redisClient`.



Tools(/dashboards/my_tools)



Video on demand(/dashboards/videos)



Peers(/users/peers)



Discord(<https://discord.com/app>)



My Profile(/users/my_profile)

```

bob@dylan:~$ cat main.js
import redisClient from './utils/redis';

(async () => {
  console.log(redisClient.isAlive());
  console.log(await redisClient.get('myKey'));
  await redisClient.set('myKey', 12, 5);
  console.log(await redisClient.get('myKey'));
  My Planning(/planning/me)
  setTimeout(async () => {
    console.log(await redisClient.get('myKey'));
  }, projects(/projects/current))
})();


✓ bob@dylan:~$ curl -s -X GET http://localhost:3000/api/courses/to_review
true
null
? 12 Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)
null
bob@dylan:~$

```


 Curriculums(/dashboards/my_curriculums)

Repo:

-  • GitHub repository: alx-files_manager
- File: utils/redis.js

 Conference rooms(/dashboards/video_rooms)

Check submission

 Get a sandbox


View results




 MongoDB (servers)

mandatory

 Score: 100.0% (Checks completed: 100.0%)

Inside the folder `utils`, create a file `db.js` that contains the class `DBClient`.

 Tools(/dashboards/my_tools)
DBClient should have:

-  • the constructor that creates a client to MongoDB:
 - host: from the environment variable `DB_HOST` or default: `localhost`
 - port: from the environment variable `DB_PORT` or default: `27017`
 - database: from the environment variable `DB_DATABASE` or default: `files_manager`
-  • a function `isAlive` that returns `true` when the connection to MongoDB is a success otherwise, `false`
- an asynchronous function `nbUsers` that returns the number of documents in the collection `users` (<https://discord.com/app>)
-  • an asynchronous function `nbFiles` that returns the number of documents in the collection `files`

After the class definition, create and export an instance of `DBClient` called `dbClient`.

My Profile(/users/my_profile)

```

bob@dylan:~$ cat main.js
import dbClient from './utils/db';


const waitConnection = () => {
  return new Promise((resolve, reject) => {
    let i = 0;
    const repeatFct = async () => {
      await setTimeout(() => {
        My Planning(/planning/me)
        Projects(/projects/current)
        else if(!dbClient.isAlive()) {
          repeatFct()
        }
        QA Reviews I can make(/corrections/to_review)
        else {
          resolve()
        }
        Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)
      }, 1000);
    };
    repeatFct();
  })
};

(async () => {
  Concepts(/concepts)
  console.log(dbClient.isAlive());
  await waitConnection();
  console.log(dbClient.isAlive());
  Conference rooms(/dashboards/video_rooms)
  console.log(await dbClient.nbUsers());
  console.log(await dbClient.nbFiles());
})();


Servers(/servers)

bob@dylan:~$ npm run dev main.js
false
true Sandboxes(/user_containers/current)
4
30
bob@dylan:~$
Tools(/dashboards/my_tools)

```

 **Repo:** Video on demand(/dashboards/videos)

- GitHub repository: alx-files_manager
- File: utils/db.js

 **Peers**(/users/peers)


Check submission

 Get a sandbox

View results

 **Discord**(<https://discord.com/app>)

2. First API

mandatory 

Score: 100.0% (Checks completed: 100.0%)

My Profile(/users/my_profile)
Inside server.js, create the Express server:

- it should listen on the port set by the environment variable `PORT` or by default 5000
- it should load all routes from the file `routes/index.js`

Inside the folder `routes`, create a file `index.js` that contains all endpoints of our API:

- GET `/status` => `AppController.getStatus`
- GET `/stats` => `AppController.getStats`

Inside the folder `controllers`, create a file `AppController.js` that contains the definition of the 2 endpoints:

- GET `/status` should return if Redis is alive and if the DB is alive too by using the 2 utils created previously: { "redis": true, "db": true } with a status code 200
- GET `/stats` should return the number of users and files in DB: { "users": 12, "files": 1231 } with a status code 200

Terminal 1: Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)

```
bob@dylan:~$ npm run start-server
Server running on port 5000
```

... Curriculums(/dashboards/my_curriculums)

Terminal 2:

```
bob@dylan:~$ curl 0.0.0.0:5000/status ; echo ""
{"redis":true,"db":true}
bob@dylan:~$ curl 0.0.0.0:5000/stats ; echo ""
{"users":4,"files":30}
```

> Repo: Sandboxes(/user_containers/current)

- GitHub repository: `alx-files_manager`
- File: `server.js`, `routes/index.js`, `controllers/AppController.js`

Video on demand(/dashboards/videos)

Check submission Get a sandbox View results

3. Create a new user

mandatory

Peers(/users/peers)

Score: 83.33% (Checks completed: 83.33%)

Discord(<https://discord.com/app>)

Now that we have a simple API, it's time to add users to our database.

In the file `routes/index.js`, add a new endpoint:

- POST `/users` => `UserController.postNew`

Inside `controllers`, add a file `UserController.js` that contains the new endpoint:

POST `/users` should create a new user in DB:

- To create a user, you must specify an email and a password
- If the email is missing, return an error Missing email with a status code 400
- If the password is missing, return an error Missing password with a status code 400
- If the email already exists in DB, return an error Already exist with a status code 400
- The password must be stored after being hashed in SHA1
- The endpoint is returning the new user with only the email and the id (auto generated by MongoDB) with a status code 201
- The new user must be saved in the collection users :
 - email : same as the value received
 - password : SHA1 value of the value received

```
bob@dylan:~$ curl 0.0.0.0:5000/users -XPOST -H "Content-Type: application/json" -d '{ "email": "bob@dylan.com", "password": "toto1234!" }' ; echo ""
{"id": "5f1e05c7b60511e683b21", "email": "bob@dylan.com"}
```

```
bob@dylan:~$
bob@dylan:~$ echo 'db.users.find()' | mongo files_manager
{ "_id" : "5f1e05c7b60511e683b21", "email" : "bob@dylan.com", "password" : "89cad29e3ebc1035b29b1478a8e70854f25fa2b2" }
```

```
bob@dylan:~$
bob@dylan:~$
bob@dylan:~$ curl 0.0.0.0:5000/users -XPOST -H "Content-Type: application/json" -d '{ "email": "bob@dylan.com", "password": "toto1234!" }' ; echo ""
{"error": "Already exist"}
```

```
bob@dylan:~$
bob@dylan:~$ curl 0.0.0.0:5000/users -XPOST -H "Content-Type: application/json" -d '{ "email": "bob@dylan.com" }' ; echo ""
{"error": "Missing password"}
```

Servers(/servers)

Repo:

- GitHub repository: alx-files_manager
- File: utils/, routes/index.js, controllers/UsersController.js

Tools(/dashboards/my_tools)

Check submission

Mark submission

Get a sandbox

View results

Authenticate a user (/dashboards/videos)

mandatory

Score: 70.0% (Checks completed: 70.0%)

Peers(/users/peers)

In the file routes/index.js , add 3 new endpoints:

- GET /connect => AuthController.getConnect
- GET /disconnect => AuthController.getDisconnect
- GET /users/me => UserController.getMe

Inside controllers , add a file AuthController.js that contains new endpoints:

GET /connect should sign-in the user by generating a new authentication token:
My Profile(/users/my_profile)



- By using the header `Authorization` and the technique of the Basic auth (Base64 of the `<email>:<password>`), find the user associate to this email and with this password (reminder: we (/) are storing the SHA1 of the password)
- if no user has been found, return an error `unauthorized` with a status code 401



- Otherwise:
 - Generate a random string (using `uuidv4`) as token
 - Create a key: `auth_<token>`



Use this key for storing in Redis (by using the `redisClient` create previously) the user ID for 24 hours



Return this token: `{ "token": "155342df-2399-41da-9e8c-458b6ac52a0c" }` with a status code 200

Now, we have a way to identify a user, create a token (= avoid to store the password on any front-end) and use this token for 24h to access to the API

Every authenticated endpoints of our API will look at this token inside the header `X-Token`.



GET /dashboards/my_base_evaluation (evaluation quizzes)

- Retrieve the user based on the token:
 - If not found, return an error `Unauthorized` with a status code 401
 - Otherwise, delete the token in Redis and return nothing with a status code 204



Curriculums(/dashboards/my_curriculums)

Inside the file `controllers/UsersController.js` add a new endpoint:



GET /users/me should retrieve the user base on the token used:

- Retrieve the user based on the token:
 - If not found, return an error `Unauthorized` with a status code 401
 - Otherwise, return the user object (email and id only)



Conference rooms(/dashboards/video_rooms)



```
bob@dylan:~$ curl 0.0.0.0:5000/connect -H "Authorization: Basic Ym9iQGR5bGFuLmNvbTp0b3RvMTIzNCE=" ; echo ""
{"token": "031bfffac-3edc-4e51-aaae-1c121317da8a"}
```



```
bob@dylan:~$ curl 0.0.0.0:5000/users/me -H "X-Token: 031bfffac-3edc-4e51-aaae-1c121317da8a" ; echo ""
{"id": "5f1e7cda04a394508232559d", "email": "bob@dylan.com"}
```



```
bob@dylan:~$ curl 0.0.0.0:5000/disconnect -H "X-Token: 031bfffac-3edc-4e51-aaae-1c121317da8a" ; echo ""
{"error": "Unauthorized"}
```



```
bob@dylan:~$ curl 0.0.0.0:5000/users/me -H "X-Token: 031bfffac-3edc-4e51-aaae-1c121317da8a" ; echo ""
{"error": "Unauthorized"}
```



```
bob@dylan:~$ curl 0.0.0.0:5000/users/peers
```



Discord(<https://discord.com/app>)

- GitHub repository: `alx-files_manager`
- File: `utils/`, `routes/index.js`, `controllers/UsersController.js`, `controllers/AuthController.js`

My Profile(/users/my_profile)



(/)

5. First file

mandatory



Home(/)

Score: 0.0% (Checks completed: 0.0%)



the file `My Planning (/planning/me)` and a new endpoint:

- `POST /files => FilesController.postUpload`



`Projects (/projects/current)`

inside `controllers`, add a file `FilesController.js` that contains the new endpoint:

`POST /files` should create a new file in DB and in disk:

`QA Reviews I can make (/corrections/to_review)`

- Retrieve the user based on the token:
 - If not found, return an error `Unauthorized` with a status code `401`



- To evaluate a file, quizzes (/classpecifics/my_current_evaluation_quizzes)

- `name` : as filename

- `type` : either `folder` , `file` or `image`

- `parentId` : (optional) as ID of the parent (default: `0` -> the root)



`Curriculum (/dashboards/my_curriculum)` if the file is public or not (default: `false`)

- `data` : (only for `type=file|image`) as Base64 of the file content



- If the `name` is missing, return an error `Missing name` with a status code `400`

`Concepts (/concepts)`

- If the `type` is missing or not part of the list of accepted type, return an error `Missing type` with a status code `400`



- If the `data` is missing and `type != folder`, return an error `Missing data` with a status code `400`

`Conference rooms (/dashboards/video_rooms)`

- If the `parentId` is set:



`Servers (/servers)`

- If no file is present in DB for this `parentId`, return an error `Parent not found` with a status code `400`



`Sandboxes (/user_containers/current)`

- If the file present in DB for this `parentId` is not of type `folder` , return an error `Parent is not a folder` with a status code `400`

- The user ID should be added to the document saved in DB - as owner of a file



- If the `type` is `folder` , add the new file document in the DB and return the new file with a status code `201`

`Tools (/dashboards/my_tools)`

- Otherwise:



`Video files will be stored locally in a folder` (to create automatically if not present):

- The relative path of this folder is given by the environment variable `FOLDER_PATH`
- If this variable is not present or empty, use `/tmp/files_manager` as storing folder path



`Peers (/peers/local)` path in the storing folder with filename a UUID

- Store the file in clear (reminder: `data` contains the Base64 of the file) in this local path

- Add the new file document in the collection `files` with these attributes:



`Discord (/https://discord.com/app)`

- `userId` : ID of the owner document (owner from the authentication)

- `name` : same as the value received

- `type` : same as the value received

- `isPublic` : same as the value received

- `parentId` : same as the value received - if not present: `0`

`My Profile (/users/my_profile)`

- `type=file|image` , the absolute path to the file save in local



- Return the new file with a status code 201



(/)



Home(/)



My Planning(/planning/me)



Projects(/projects/current)



QA Reviews I can make(/corrections/to_review)



Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)



Curriculums(/dashboards/my_curriculums)



Concepts(/concepts)



Conference rooms(/dashboards/video_rooms)



Servers(/servers)



Sandboxes(/user_containers/current)



Tools(/dashboards/my_tools)



Video on demand(/dashboards/videos)



Peers(/users/peers)



Discord(<https://discord.com/app>)



My Profile(/users/my_profile)

```

bob@dylan:~$ curl 0.0.0.0:5000/connect -H "Authorization: Basic Ym9iQGR5bGFuLmNvb
Tp0b3RVMTIzNCE=" ; echo ""
{"token": "f21fb953-16f9-46ed-8d9c-84c6450ec80f"}
bob@dylan:~$
bob@dylan:~$ curl -XPOST 0.0.0.0:5000/files -H "X-Token: f21fb953-16f9-46ed-8d9c-
84c6450ec80f" -H "Content-Type: application/json" -d '{"name": "myText.txt", "ty
pe": "file", "data": "SGVsbG8gV2Vic3RhY2shCg==" }' ; echo ""
{"id": "5f1e879ec7ba06511e683b22", "userId": "5f1e7cda04a394508232559d", "name": "myTe
xt.txt", "type": "file", "isPublic": false, "parentId": 0}
bob@dylan:~$
bob@dylan:~$ ls /tmp/files_manager/
2a1f4fc3-687b-491a-a3d2-5808a02942c9
bob@dylan:~$ cat /tmp/files_manager/2a1f4fc3-687b-491a-a3d2-5808a02942c9
Hello World
bob@dylan:~$
bob@dylan:~$ curl -XPOST 0.0.0.0:5000/files -H "X-Token: f21fb953-16f9-46ed-8d9c-
84c6450ec80f" -H "Content-Type: application/json" -d '{"name": "images", "type":
"folder" }' ; echo ""
{"id": "5f1e881cc7ba06511e683b23", "userId": "5f1e7cda04a394508232559d", "name": "imag
es", "type": "folder", "isPublic": false, "parentId": 0}
bob@dylan:~$
bob@dylan:~$ cat image_upload.py
import base64
import requests
import sys

file_path = sys.argv[1]
file_name = file_path.split("/")[-1]

file_encoded = None
with open(file_path, "rb") as image_file:
    file_encoded = base64.b64encode(image_file.read()).decode('utf-8')

r_json = {
    "name": file_name,
    "type": "image",
    "isPublic": True,
    "data": file_enc
oded,
    "parentId": sys.argv[3]
}
r_headers = {
    "X-Token": sys.argv[2]
}
r = requests.post("http://0.0.0.0:5000/files", json=r_json, headers=r_headers)
print(r.json())

bob@dylan:~$
bob@dylan:~$ python image_upload.py image.png f21fb953-16f9-46ed-8d9c-84c6450ec80
f 5f1e881cc7ba06511e683b23
{"id": "5f1e8896c7ba06511e683b25", "userId": "5f1e7cda04a394508232559d", "name":
"image.png", "type": "image", "isPublic": True, "parentId": "5f1e881cc7ba06511e68
3b23"}
bob@dylan:~$
bob@dylan:~$ echo 'mongo files_manager
{ "_id" : ObjectId("5f1e881cc7ba06511e683b23"), "userId" : ObjectId("5f1e7cda04a3
94508232559d"), "name" : "images", "type" : "folder", "parentId" : "0" }
{ "_id" : ObjectId("5f1e879ec7ba06511e683b22"), "userId" : ObjectId("5f1e7cda04a3
94508232559d"), "name" : "myText.txt", "type" : "file", "parentId" : "0", "isPubl
ic" : false, "localPath" : "/tmp/files_manager/2a1f4fc3-687b-491a-a3d2-5808a02942
c9" }'

```

```
{ "_id" : ObjectId("5f1e8896c7ba06511e683b25"), "userId" : ObjectId("5f1e7cda04a394508232559d"), "name" : "image.png", "type" : "image", "parentId" : ObjectId("5f1e881cc7ba06511e683b23"), "isPublic" : true, "localPath" : "/tmp/files_manager/51997b88-5c42-42c2-901e-e7f4e71bdc47" }
```

bob@dylan:~\$

bob@dylan:~\$ ls /tmp/files_manager/
2a1f4fc3-687b-491a-a3d2-5808a02942c9 51997b88-5c42-42c2-901e-e7f4e71bdc47

bob@dylan:~\$



My Planning(/planning/me)



Repo:

Projects(/projects/current)

- GitHub repository: alx-files_manager
 - File: utils/, routes/index.js, controllers/FilesController.js
- ✓ QA Reviews I can make(/corrections/to_review)

Check submission

Mark submission

> Get a sandbox

View results



Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)

6. Get and list file

mandatory



Score: 0.0% (Checks completed: 0.0%)
Curriculums(/dashboards/my_curriculums)

In the file routes/index.js, add 2 new endpoints:



Concepts(/concepts)

- GET /files/:id => FilesController.getShow
- GET /files => FilesController.getIndex



Conference rooms(/dashboards/video_rooms)

In the file controllers/FilesController.js, add the 2 new endpoints:

GET /files/:id should retrieve the file document based on the ID:



Servers(/servers)

- Retrieve the user based on the token:
 - If not found, return an error Unauthorized with a status code 401
- If a file does not exist (not linked to the user) and the ID passed as parameter, return an error Not found with a status code 404
- Otherwise, return the file document



Tools(/dashboards/my_tools)

GET /files should retrieve all users file documents for a specific parentId and with pagination:



Retrieve the user based on the token:
Video on demand(/dashboards/videos)

- If not found, return an error Unauthorized with a status code 401
- Based on the query parameters parentId and page, return the list of file document
 - parentId:



Peers(/users/peers)

- No validation of parentId needed - if the parentId is not linked to any user folder, returns an empty list

- By default, parentId is equal to 0 = the root



Discord pagination(/discord.com/app)

- Each page should be 20 items max
- page query parameter starts at 0 for the first page. If equals to 1, it means it's the second page (from the 20th to the 40th), etc...
- Pagination can be done directly by the aggregate of MongoDB

My Profile(/users/my_profile)



```

bob@dylan:~$ curl 0.0.0.0:5000/connect -H "Authorization: Basic Ym9iQGR5bGFuLmNvb
Tp0b3RVMTIzNCE=" ; echo ""
{"token":"f21fb953-16f9-46ed-8d9c-84c6450ec80f"}
bob@dylan:~$
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files -H "X-Token: f21fb953-16f9-46ed-8d9c-8
4c6450ec80f" ; echo ""
[{"id":"5f1e879ec7ba06511e683b22","userId":"5f1e7cda04a394508232559d","name":"myT
ext.txt","type":"file","isPublic":false,"parentId":0},{id":"5f1e881cc7ba06511e68
3b23","name":"My Planning (planning/me
My Planning (planning/me)
Projects/projects/current)
bob@dylan:~$
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files?parentId=5f1e881cc7ba06511e683b23 -H
"X-Token: f21fb953-16f9-46ed-8d9c-84c6450ec80f" ; echo ""
[{"id":"5f1e8896c7ba06511e683b25","userId":"5f1e7cda04a394508232559d","name":"ima
ge.png","type":"image","isPublic":true,"parentId":"5f1e881cc7ba06511e683b23"}]
bob@dylan:~$
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files/5f1e8896c7ba06511e683b25 -H "X-Token:
f21fb953-16f9-46ed-8d9c-84c6450ec80f" ; echo ""
{"id":"5f1e8896c7ba06511e683b25","userId":"5f1e7cda04a394508232559d","name":"imag
e.png","type":"image","isPublic":true,"parentId":"5f1e881cc7ba06511e683b23"}
bob@dylan:~$

```



Concepts(/concepts)

Repo:



- GitHub repository: alx-files_manager
- Conference rooms(/dashboards/video_rooms)
- File: utils/, routes/index.js, controllers/FilesController.js



Servers(/servers)

Check submission

Mark submission

> Get a sandbox

View results



File publish/unpublish containers/current)

mandatory



Score: 0.0% (Checks completed: 0.0%)

In the file routes/index.js , add 2 new endpoints:



Video on demand(/dashboards/videos)

- PUT /files/:id/publish => FilesController.putPublish
- PUT /files/:id/publish => FilesController.putUnpublish



In the file controllers/FilesController.js , add the 2 new endpoints:

Peers(/users/peers)

PUT /files/:id/publish should set isPublic to true on the file document based on the ID:



- Retrieve the user based on the token:
 - Discord(<https://discord.com/app>)
 - If not found, return an error Unauthorized with a status code 401
- If no file document is linked to the user and the ID passed as parameter, return an error Not found with a status code 404
- Otherwise:
 - Update the value of isPublic to true
 - And return the file document with a status code 200



PUT /files/:id/unpublish should set isPublic to false on the file document based on the ID:



- Retrieve the user based on the token:
 - (/) ◦ If not found, return an error Unauthorized with a status code 401
- If no file document is linked to the user and the ID passed as parameter, return an error Not found with a status code 404
- Otherwise:
 - Update the value of isPublic to false
 - And return the file document with a status code 200



```
bob@dylan:~$ curl 0.0.0.0:5000/connect -H "Authorization: Basic Ym9iQGR5bGFuLmNvbTpoY3RVM4I2NCE=" ; echo
{"token": "f21fb953-16f9-46ed-8d9c-84c6450ec80f"}
bob@dylan:~$
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files/5f1e8896c7ba06511e683b25 -H "X-Token: f21fb953-16f9-46ed-8d9c-84c6450ec80f" ; echo ""
{"id": "5f1e8896c7ba06511e683b25", "userId": "5f1e7cda04a394508232559d", "name": "image.png", "type": "image", "isPublic": true, "parentId": "5f1e881cc7ba06511e683b23"}
bob@dylan:~$
bob@dylan:~$ curl -XPUT 0.0.0.0:5000/files/5f1e8896c7ba06511e683b25/publish -H "X-Token: f21fb953-16f9-46ed-8d9c-84c6450ec80f" ; echo ""
{"id": "5f1e8896c7ba06511e683b25", "userId": "5f1e7cda04a394508232559d", "name": "image.png", "type": "image", "isPublic": true, "parentId": "5f1e881cc7ba06511e683b23"}
bob@dylan:~$
bob@dylan:~$ curl -XPUT 0.0.0.0:5000/files/5f1e8896c7ba06511e683b25/unpublish -H "X-Token: f21fb953-16f9-46ed-8d9c-84c6450ec80f" ; echo ""
{"id": "5f1e8896c7ba06511e683b25", "userId": "5f1e7cda04a394508232559d", "name": "image.png", "type": "image", "isPublic": false, "parentId": "5f1e881cc7ba06511e683b23"}
bob@dylan:~$
```



Servers(/servers)

Repo:



- GitHub repository: alx-files_manager
- File: utils/, routes/index.js, controllers/FilesController.js



Tools(/dashboards/my_tools)

Check submission

Mark submission

Get a sandbox

View results



Video on demand(/dashboards/videos)

mandatory

Score: 0.0% (Checks completed: 0.0%)



Peers(/users/peers)

In the file routes/index.js , add one new endpoint:



- GET /files/:id/data => FilesController.getFile

Discord(https://discord.com/app)

In the file controllers/FilesController.js , add the new endpoint:

GET /files/:id/data should return the content of the file document based on the ID:

- If no file document is linked to the ID passed as parameter, return an error Not found with a status code 404

My Profile(/users/my_profile)





- If the file document (folder or file) is not public (isPublic: false) and no user authenticate or not the owner of the file, return an error Not found with a status code 404
- If the type of the file document is folder ,return an error A folder doesn't have content with a status code 400



- If the file is not locally present, return an error Not found with a status code 404
- Otherwise:



- By using the module mime-types ,get the MIME-type
- Return the content of the file with the correct MIME-type



```
bob@dylan:~$ curl -XPUT 0.0.0.0:5000/connect -H "Authorization: Basic Ym9iQGR5bGFuLmNvbTp0b3RVMTIzNCE=" ; echo ""
{"token": "f21fb953-16f9-46ed-8d9c-84c6450ec80f"}
```



```
bob@dylan:~$ curl -XPUT 0.0.0.0:5000/files/5f1e879ec7ba06511e683b22/unpublish -H "X-Token: f21fb953-16f9-46ed-8d9c-84c6450ec80f" ; echo ""
```



```
{"id": "5f1e879ec7ba06511e683b22", "userId": "5f1e7cda04a394508232559d", "name": "myText.txt", "type": "file", "isPublic": false, "parentId": 0}
```



```
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files/5f1e879ec7ba06511e683b22/data -H "X-Token: f21fb953-16f9-46ed-8d9c-84c6450ec80f" ; echo ""
Hello Webstack!
```



```
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files/5f1e879ec7ba06511e683b22/data ; echo ""
{"error": "Not found"}
```



```
bob@dylan:~$ curl -XPUT 0.0.0.0:5000/dashboards/videos/5f1e879ec7ba06511e683b22/publish -H "X-Token: f21fb953-16f9-46ed-8d9c-84c6450ec80f" ; echo ""
{"id": "5f1e879ec7ba06511e683b22", "userId": "5f1e7cda04a394508232559d", "name": "myText.txt", "type": "video", "isPublic": true, "parentId": 0}
```



```
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files/5f1e879ec7ba06511e683b22/data ; echo ""
Sandboxes(/user_containers/current)
Hello Webstack!
```



```
bob@dylan:~$ curl -XGET 0.0.0.0:5000/dashboards/my_tools
```



Repo: Video on demand(/dashboards/videos)

- GitHub repository: alx-files_manager
- File: utils/, routes/index.js, controllers/FilesController.js



Peers(/users/peers)

Check submission

Mark submission

View results



Discord(<https://discord.com/app>)

9. Image Thumbnails

mandatory

Score: 0.0% (Checks completed: 0.0%)

My Profile(/users/my_profile)

Update the endpoint `POST /files` endpoint to start a background processing for generating thumbnails for a file of type `image` :

(/) Create a Bull queue `fileQueue`

- When a new image is stored (in local and in DB), add a job to this queue with the `userId` and `fileId`

Create a file `worker.js` :

- By using the module `Bull`, create a queue `fileQueue`
- Process this queue:
 - If `fileId` is not present in the job, raise an error `Missing fileId`
 - If `userId` is not present in the job, raise an error `Missing userId`
 - If no document is found in DB based on the `fileId` and `userId`, raise an error `File not found`
 - By using the module `image-thumbnail`, generate 3 thumbnails with `width = 500, 250 and 100` - store each result on the same location of the original file by appending `_width size`

Update the endpoint `GET /files/:id/data` to accept a query parameter `size` :

- `size` can be `500, 250 or 100`
- Based on `size`, return the correct local file
- If the local file doesn't exist, return an error `Not found` with a status code `404`

Terminal 3: (start the worker)

```
bob@dylan:~$ npm run start-worker
...
```

Terminal 2: Servers(/servers)

Sandboxes(/user_containers/current)

Tools(/dashboards/my_tools)

Video on demand(/dashboards/videos)

Peers(/users/peers)

Discord(<https://discord.com/app>)

My Profile(/users/my_profile)




```

bob@dylan:~$ curl 0.0.0.0:5000/connect -H "Authorization: Basic Ym9iQGR5bGFuLmNvb
Tp0b3RVMTIzNCE=" ; echo ""
{"token": "f21fb953-16f9-46ed-8d9c-84c6450ec80f"}
bob@dylan:~$
bob@dylan:~$ python image_upload.py image.png f21fb953-16f9-46ed-8d9c-84c6450ec80
f5f1e881cc7ba06511e683b23
{'id': '5f1e8896c7ba06511e683b25', 'userId': '5f1e7cda04a394508232559d', 'name':
'image.png', 'type': 'image', 'isPublic': True, 'parentId': '5f1e881cc7ba06511e68
3b23'}
My Planning(/planning/me)
bob@dylan:~$ ls /tmp/files_manager/
2a1f4fc3-687b-491a-a3d2-5808a02942c9    51997b88-5c42-42c2-901e-e7f4e71bdc47    6dc
53397-8491-4b7c-8273-f748b1a031cb    6dc53397-8491-4b7c-8273-f748b1a031cb_100    6d
c53397-8491-4b7c-8273-f748b1a031cb_250    6dc53397-8491-4b7c-8273-f748b1a031cb_50
0
bob@dylan:~$
QA Reviews I can make(/corrections/to_review)
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files/5f1e8896c7ba06511e683b25/data -so new_
image.png ; file new_image.png
new_image.png: PNG image data, 100 x 109, 8-bit/color RGBA, non-interlaced
bob@dylan:~$
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files/5f1e8896c7ba06511e683b25/data?size=100
-so new_image.png ; file new_image.png
new_image.png: PNG image data, 100 x 109, 8-bit/color RGBA, non-interlaced
bob@dylan:~$
bob@dylan:~$ curl -XGET 0.0.0.0:5000/files/5f1e8896c7ba06511e683b25/data?size=250
-so new_image.png ; file new_image.png
new_image.png: PNG image data, 250 x 272, 8-bit/color RGBA, non-interlaced
bob@dylan:~$

```

Conference rooms(/dashboards/video_rooms)

Repo:

- Servers(/servers)
- GitHub repository: alx-files_manager
- File: utils/, controllers/FilesController.js, worker.js

Sandboxes(/user_containers/current)

View results

Tools(/dashboards/my_tools)

10. Tests!

#advanced

Video on demand(/dashboards/videos)
Score: 0.0% (Checks completed: 0.0%)

Of course, a strong and stable project can not be good without tests.

Peers(/users/peers)
Create tests for each endpoint and dbClient.

Create tests for each endpoints:

- Discord(<https://discord.com/app>)
- GET /status
- GET /stats
- POST /users
- GET /connect
- GET /disconnect
- My Profile(/users/my_profile)
- GET /users/me



- POST /files
- GET /files/:id
- GET /files (don't forget the pagination)
- PUT /files/:id/publish
- PUT /files/:id/unpublish
- GET /files/:id/data

Repo: My Planning(/planning/me)

- GitHub repository: alx-files_manager
- Projects(/projects/current)
- File: tests/

✓ QA Reviews I can make(/corrections/to_review)
View results

? 1. New user welcome email Evaluation boards/my_current_evaluation_quizzes)

#advanced

Score: 0.0% (Checks completed: 0.0%)

Curriculums(/dashboards/my_curriculums)
Update the endpoint POST /users endpoint to start a background processing for sending a "Welcome email" to the user:

- Create a Bull queue userQueue
- When a new user is stored (in DB), add a job to this queue with the userId

Conference rooms(/dashboards/video_rooms)
Update the endpoint POST /users

- By using the module Bull, create a queue userQueue
- Process this queue:
 - If userId is not present in the job, raise an error Missing userId
 - If no document is found in DB based on the userId, raise an error User not found

Sandboxes(/user_containers/current)
Print in the console Welcome <email>!

In real life, you can use a third party service like Mailgun (/rtoken/Lr8MoeTgC7KRx413wpoBXg) to send real emails. (Using an API on a server via SMTP is worst!) and sending emails via a background job is important to optimize API endpoint.

Video on demand(/dashboards/videos)
Repo:

- GitHub repository: alx-files_manager
- File: utils/, worker.js, controllers/UsersController.js
- Peers(/users/peers)

View results
Discord(https://discord.com/app)



Ready for a new review

My Profile(/users/my_profile)



(/)

Copyright © 2024 ALX, All rights reserved.



Home(/)



My Planning(/planning/me)



Projects(/projects/current)



QA Reviews I can make(/corrections/to_review)



Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)



Curriculums(/dashboards/my_curriculums)



Concepts(/concepts)



Conference rooms(/dashboards/video_rooms)



Servers(/servers)



Sandboxes(/user_containers/current)



Tools(/dashboards/my_tools)



Video on demand(/dashboards/videos)



Peers(/users/peers)



Discord(<https://discord.com/app>)



My Profile(/users/my_profile)