# **SETUP**

1. Clonar repositorio: <https://github.com/vfJorge/Histolok-UADY/>

## **SETUP BACKEND**

1. Ir a ./Histolok-backend/
2. composer update
3. php artisan storage:link

DEV:

1. php artisan serve
2. Si se quiere actualizar la estructura de la db: php artisan migrate

## **SETUP FRONT (Xampp)**

1. Ir a ./histolok/
2. npm install
3. npm run build
4. Copiar la carpeta ./histolok/dist/histolok
5. Pegarla en ./xampp/htdocs/
6. Ver que el mod\_rewrite este activo en httpd.conf (Xampp)
7. Incluir archivo .htaccess:

<IfModule mod\_rewrite.c>

RewriteEngine on

# Don't rewrite files or directories

RewriteCond %{REQUEST\_FILENAME} -f [OR]

RewriteCond %{REQUEST\_FILENAME} -d

RewriteRule ^ - [L]

# Rewrite everything else to index.html

# to allow html5 state links

RewriteRule ^ index.html [L]

</IfModule>

DEV:

1. ng serve

## **SETUP XAMPP**

1. Correr Apache + MySQL
2. Crear DB histolokdb
3. Importar histolokdb.sql

## **RUTAS BACKEND**

TODAS LAS RUTAS ESTAN EN **Histolok-backend\routes\api.php**

1. Q ASE
2. RUTA
3. METODO (opcional)
4. CAMPOS

**RUTAS SIN BEARER**

**Registro**

http://127.0.0.1:8000/api/auth/register

POST

name, email, password, password\_confirmation

**Login**

http://127.0.0.1:8000/api/auth/login

POST

email, password

**RUTAS CON BEARER**

**Log out**

http://127.0.0.1:8000/api/auth/logout

POST

**Ver tipo de usuario**

http://127.0.0.1:8000/api/auth/type

GET

**USERS**

**Ver todos**

http://127.0.0.1:8000/api/users

GET

**Ver info de perfil**

http://127.0.0.1:8000/api/users/me

GET

**Ver info de cualquier usuario**

http://127.0.0.1:8000/api/users/1

GET

**Cambiar info de perfil (solo admin puede cambiar cuentas ajenas)**

http://127.0.0.1:8000/api/users/1

PUT

Name, type (type solo funciona si eres admin)(type = A,S,E) S de superusuario en vez de profesor xd

**Borrar cuenta**

http://127.0.0.1:8000/api/users/1

DELETE

**Ver resultados de todos los exámenes presentados NO IMPLEMENTADO AUN**

http://127.0.0.1:8000/api/users/1/results  
GET

**FOTOS**

**Ver todas las fotos (SOLO ADMIN)**

http://127.0.0.1:8000/api/fotos

GET

**Ver todas las fotos publicas (PROFES Y ADMIN)**

http://127.0.0.1:8000/api/fotos/public

GET

**Ver fotos subidas por el usuario**

http://127.0.0.1:8000/api/fotos/me es para ver las fotos que pertenecen al usuario

GET

**Crear foto**

http://127.0.0.1:8000/api/fotos

POST

title, desc, keywords(array, ej. ["corazon","celula","epitelio"]), image(file)

(necesitas poner Content-type= multipart/form-data)

**Ver info de una foto**

http://127.0.0.1:8000/api/fotos/1

GET

**Cambiar foto (solo el usuario que la creo puede)**

http://127.0.0.1:8000/api/fotos/1?\_method=PUT (no se puede mandar PUT pq laravel no acepta el multipart/form-data, pero en POST si, pero con lo del query se arregla)

POST

title, desc, keywords (array, ej. ["corazon","celula","epitelio"]), image(file)

(necesitas poner Content-type= multipart/form-data)

**Borrar foto (solo el creador puede)**

http://127.0.0.1:8000/api/fotos/1

DELETE

**PREGUNTAS**

**Ver todas las preguntas (SOLO ADMIN)**

http://127.0.0.1:8000/api/preguntas

GET

**Ver todas las preguntas públicas (PROFES Y ADMIN)**

http://127.0.0.1:8000/api/fotos/preguntas

GET

**Ver preguntas subidas por el usuario**

http://127.0.0.1:8000/api/preguntas/me es para ver las fotos que pertenecen al usuario

GET

**Crear pregunta**

http://127.0.0.1:8000/api/preguntas

POST

title, question, keywords(array, ej. ["corazon","celula","epitelio"]), answer(required), option1(required), option2, option3, option4, access(public/private), difficulty(1-5), foto\_id(debe existir la foto y ser publica)

PD: answer(la respuesta) es aparte de las opciones, es decir, puede haber hasta 5 opciones.

**Ver info de una pregunta**

http://127.0.0.1:8000/api/preguntas/1

GET

**Cambiar pregunta (solo el usuario que la creo puede)**

http://127.0.0.1:8000/api/preguntas/1

PUT

title, question, keywords(array, ej. ["corazon","celula","epitelio"]), answer(required), option1(required), option2, option3, option4, access(public/private), difficulty(1-5),

**Borrar pregunta (solo el creador puede)**

http://127.0.0.1:8000/api/preguntas/1

DELETE

**CRUD EXAMENES (NUEVO FASE 3)**

**Ver todas los examenes (SOLO ADMIN)**

GET

http://127.0.0.1:8000/api/examenes

**Ver todas los examenes públicas (PROFES Y ADMIN)**

GET

http://127.0.0.1:8000/api/examenes/public

**Ver los examenes creados por el usuario**

GET

http://127.0.0.1:8000/api/examenes/me es para ver los examenes que creó el usuario

**Crear examen**

POST

http://127.0.0.1:8000/api/examenes

title, description, keywords(array, ej. ["corazon","celula","epitelio"]), access(public/private), duration(00:00:00), difficulty(1-5), questions (array, ej. ["1","3","4",”7”,”2”,”5”])

**Generar examen (dinámico) NUEVO** [✨](https://emojipedia.org/es/chispas/) **[ALUMNO NO IMPLEMENTADO AUN](https://emojipedia.org/es/chispas/)**

POST

http://127.0.0.1:8000/api/examenes/generate

title, keywords(array, ej. ["corazon","celula","epitelio"]) (**opcional**), duration(00:00:00), difficulty(1-5), n\_questions (int)

**PD: NO MODIFICABLE, SI NO HAY SUFICIENTES PREGUNTAS SE GENERA EL EXAMEN CON LAS QUE HAYA, PONER KEYWORDS ES OPCIONAL.**

**Ver un examen con detalles (las preguntas y sus opciones)**

GET

http://127.0.0.1:8000/api/examenes/3

**Cambiar examen (solo el usuario que lo creo puede)**

PUT

http://127.0.0.1:8000/api/examenes/3

title, description, keywords(array, ej. ["corazon","celula","epitelio"]), access(public/private), duration(00:00:00), difficulty(1-5), questions (array, ej. ["1","3","4",”7”,”2”,”5”])

**Borrar examen (solo el creador puede)**

DELETE

http://127.0.0.1:8000/api/examenes/4

**PRESENTAR EXAMENES (NUEVO FASE 4) ALUMNO**

**Empezar examen**

GET

http://127.0.0.1:8000/api/examenes/3/start

RETORNA LA PRIMERA PREGUNTA DEL EXAMEN e inicia el temporizador.

**Pregunta actual del examen**

POST

http://127.0.0.1:8000/api/examenes/3/current

RETORNA CUANTAS PREGUNTAS QUEDAN

Y LA PREGUNTA ACTUAL

**Siguiente pregunta del examen**

POST

http://127.0.0.1:8000/api/examenes/3/next

RETORNA EL ID DE LA RESPUESTA ANTERIOR, Y

LA SIGUIENTE PREGUNTA DEL EXAMEN

\*EN CASO DE SER LA ULTIMA PREGUNTA, REGRESA “fin examen”

**Resultados ultimo examen contestado (CACHE) (es decir, no se queda para siempre creo)**

POST

http://127.0.0.1:8000/api/examenes/3/results

RETORNA LOS RESULTADOS DEL EXAMEN Y LOS ID DE LAS RESPUESTAS Y LAS OPCIONES SELECCIONADAS

**GRUPOS () NO IMPLEMENTADO AUN**

**Ver todos los grupos**

http://127.0.0.1:8000/api/grupos (solo admin)

GET

**Ver todos los grupos a los que pertenece el usuario**

http://127.0.0.1:8000/api/grupos/me es para ver los grupos en el que está el usuario

GET

**Crear grupo**

http://127.0.0.1:8000/api/grupos

POST

name,desc, users(array, ej.([“1”,”2”,”3”]));

**Ver info de grupo**

http://127.0.0.1:8000/api/grupos/1

GET

**Cambiar grupo**

http://127.0.0.1:8000/api/grupos

PUT

name,desc, users(array, ej.([“1”,”2”,”3”]));

**Borrar grupo**

http://127.0.0.1:8000/api/grupos/1

DELETE