

Pruebas de fuego DevLocker V1

1) Registrar user A y user B

Inicializamos proyecto con npm run dev

```
PS D:\DOCUMENTOS USUARIO\Documents\backend_reto> npm run dev

> devlocker-v1@1.0.0 dev
> nodemon server.js

[nodemon] 3.1.14
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node server.js`
Server running on port 3000
```

Abrimos Thunder Client y creamos el primer usuario (User A) para obtener credenciales válidas y luego iniciar sesión.

En el body ponemos el user y la contraseña del usuario en formato JSON

The screenshot shows the Thunder Client interface. On the left, there's a sidebar with activity logs for a POST request to 'localhost:3000/api/v1/registro' made 'just now'. The main area shows a POST request to 'http://localhost:3000/api/v1/registro'. The 'Body' tab is selected, displaying a JSON object:

```
1 {
2   "id": "699fe554f6543a5f4d5f6310",
3   "email": "usera@test.com"
4 }
```

The response panel shows the status: 201 Created, Size: 58 Bytes, Time: 164 ms. The response body is identical to the one shown in the body tab.

Viewing Collection: users

The screenshot shows a MongoDB interface with a search bar at the top. Below the search bar, there is a red banner with the text "Delete all 1 documents retrieved". The main area displays a table with one document:

_id	email	password	createdAt	updatedAt	__v
699fe554f6543a5f4d5f6310	usera@test.com	\$2a\$10\$8ePVHI0TZbKT7EoE0rDoJ.MSIq2rR759yPB7D7Hean...	Thu Feb 26 2026 06:16:52 GMT+0000 (Coordinated Universal Time)	Thu Feb 26 2026 06:16:52 GMT+0000 (Coordinated Universal Time)	0

Obtuvimos como resultado un status 201 por lo tanto se recibió id y email del usuario A y quedó creado.

Mismo proceso con user B

The screenshot shows the Thunder Client interface. On the left, there is a sidebar with activity logs. In the center, a new request is being prepared:

- Method: POST
- URL: http://localhost:3000/api/v1/registro
- Body:

```
1 {
2   "email": "userb@test.com",
3   "password": "12345678"
4 }
```

The response on the right shows a 201 Created status with the following JSON content:

```
1 {
2   "id": "699fe785f6543a5f4d5f6313",
3   "email": "userb@test.com"
4 }
```

Viewing Collection: users

The screenshot shows a MongoDB interface with a search bar at the top. Below the search bar, there is a red banner with the text "Delete all 2 documents retrieved". The main area displays a table with two documents:

_id	email	password	createdAt	updatedAt	__v
699fe554f6543a5f4d5f6310	usera@test.com	\$2a\$10\$8ePVHI0TZbKT7EoE0rDoJ.MSIq2rR759yPB7D7Hean...	Thu Feb 26 2026 06:16:52 GMT+0000 (Coordinated Universal Time)	Thu Feb 26 2026 06:16:52 GMT+0000 (Coordinated Universal Time)	0
699fe785f6543a5f4d5f6313	userb@test.com	\$2a\$10\$CNJmgwOsgBqUr9VV.WYnOQk0Ggu/2pwgdegVtiKtF...	Thu Feb 26 2026 06:26:13 GMT+0000 (Coordinated Universal Time)	Thu Feb 26 2026 06:26:13 GMT+0000 (Coordinated Universal Time)	0

2) Login con los users para obtener token jwt para que podamos acceder a los endpoint protegidos:

The screenshot shows the Postman interface with a successful API call. The request method is POST to `http://localhost:3000/api/v1/login`. The response status is **200 OK**, size is 183 Bytes, and time is 99 ms. The response body contains a JSON object with a single key `token` containing a long JWT string.

```

1 {
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY5OWZlNTU0ZjY1NDNhNWY0ZDVMNjMxMCIsImhlhdCI6MTc3MjA4NzQ4MiwiZXhwIjoxNzcyMTczODgyfQ.6gtowJ1n6_ty6eGBhsCwboyctGImUIZIKCVEHuRfwZo"
3 }

```

This screenshot shows another successful login request in Postman. The request method is POST to `http://localhost:3000/api/v1/login`. The response status is **200 OK**, size is 183 Bytes, and time is 92 ms. The response body contains a JSON object with a single key `token` containing a different long JWT string.

```

1 {
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY5OWZlNzg1ZjY1NDNhNWY0ZDVMNjMxMyIsImhlhdCI6MTc3MjA4NzU3OSw1ZXhwIjoxNzcyMTczOTc5fQ.tYoZ94jG_f4I4gYM68LP_4MeKDtY_u917kr1Jfyb6GA"
3 }

```

la autenticación fue exitosa y obtuvimos los tokens de user A y B

3) Creación de Snippet con user A

The screenshot shows a new Postman request to `http://localhost:3000/api/v1/snippets`. The `Auth` tab is selected, and the `Bearer` token input field contains the JWT token obtained from user A. The response area is currently empty.

Con el método POST en Auth->Bearer ponemos el token del user A y luego en Body procedemos a poner el snippet

File Edit Selection View Go Run ...

backend reto [Administrator]

EXPLORER

BACKEND RETO

- > node_modules
- src
- controllers
 - authController.js
 - snippetsController.js
- middleware
 - auth.js
 - errorHandler.js
- models
 - Snippet.js
 - User.js
- routes
 - authRoutes.js

TC New Request TC New Request TC New Request X

POST http://localhost:3000/api/v1/snippets Send

Query Headers 2 Auth 1 Body 1 Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

```

1  {
2    "title": "Snippet de User A (QA)",
3    "language": "javascript",
4    "code": "console.log('Hola desde User A');"
5  }

```

Status: 201 Created Size: 264 Bytes Time: 21 ms

Response Headers 7 Cookies Results Docs { } ≡

```

1  {
2    "user": "699fe554f6543a5f4d5f6310",
3    "title": "Snippet de User A (QA)",
4    "language": "javascript",
5    "code": "console.log('Hola desde User A');",
6    "tags": [],
7    "_id": "699feb4bf6543a5f4d5f6317",
8    "createdAt": "2026-02-26T06:42:19.539Z",
9    "updatedAt": "2026-02-26T06:42:19.539Z",
10   "__v": 0
11 }

```

- 4) Se intenta Borrar el snippet creado por el usuario A por el usuario B usando el token del usuario B:

File Edit Selection View Go Run ...

backend reto [Administrator]

THUNDER CLIENT

New Request

Activity Collections Env

filter activity

DEL localhost:3000/api/v1/snippets/699feb... just now

POST localhost:3000/api/v1/snippets 5 mins ago

POST localhost:3000/api/v1/login 15 mins ago

TC New Request TC New Request TC New Request * TC New Request X

DELETE http://localhost:3000/api/v1/snippets/699feb4bf6543a5f4d5f6317 Send

Query Headers 2 Auth 1 Body Tests Pre Run

None Basic Bearer OAuth 2 NTLM AWS

Bearer Token

```

1  {
2    "error": "Snippet no encontrado"
3  }

```

Status: 404 Not Found Size: 33 Bytes Time: 12 ms

Response Headers 7 Cookies Results Docs { } ≡

Obtenemos como resultado snippet no encontrado, eso significa que funciona correctamente y si hay privacidad, el snippet sigue estando.

Mongo Express Database: devlocker > Collection: snippets

[New Document](#) [New Index](#)

[Simple](#) [Advanced](#)

Key Value String [Find](#)

[Delete all 1 documents retrieved](#)

_id	user	title	language	code	tags	createdAt	updatedAt	__v
 699fe554f6543a5f4d5f6317	699fe554f6543a5f4d5f6310	Snippet de User A (QA)	javascript	console.log('Hola desde User A');		Thu Feb 26 2026 06:42:19 GMT+0000 (Coordinated Universal Time)	Thu Feb 26 2026 06:42:19 GMT+0000 (Coordinated Universal Time)	0

Si usamos el token A, que si esta autorizado:

File Edit Selection View Go Run ...

THUNDER CLIENT

New Request

Activity Collections Env

filter activity

DEL localhost:3000/api/v1/snippets/699feb... just now

POST localhost:3000/api/v1/snippets 5 mins ago

POST localhost:3000/api/v1/login 15 mins ago

backend reto [Administrator]

TC New Request TC New Request * TC New Request X

DELETE http://localhost:3000/api/v1/snippets/699feb4bf6543a5f4d5f6310 Send

Query Headers 2 Auth 1 Body Tests Pre Run

None Basic Bearer OAuth 2 NTLM AWS

Status: 200 OK Size: 31 Bytes Time: 10 ms

Response Headers 7 Cookies Results Docs

```

1 {
2   "mensaje": "Snippet eliminado"
3 }
```

Bearer Token

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY5OWZINTU0ZjY1NDNhNWY0ZDVmNjMxMCIsImIhdCI6MTc3MjA4NzQ4MiwiZXhwIjoxNzcyMTczODgyfQ.6gtowJ1n6_ty6eGBhsCWboycGlmUIZIKCVEHuRfWz0

Mongo Express Database: devlocker > Collection: snippets

Viewing Collection: snippets

[New Document](#) [New Index](#)

[Simple](#) [Advanced](#)

Key Value String [Find](#)

No documents found.

Se elimina correctamente