

Hito 1 - Aplicación Backend en Node y Express

Alumno: Nicolás Chávez Garzón

Grupo: 1

1. **Generar el proyecto en Node JS con la instalación de las dependencias Express, JWT y Nodemon.**

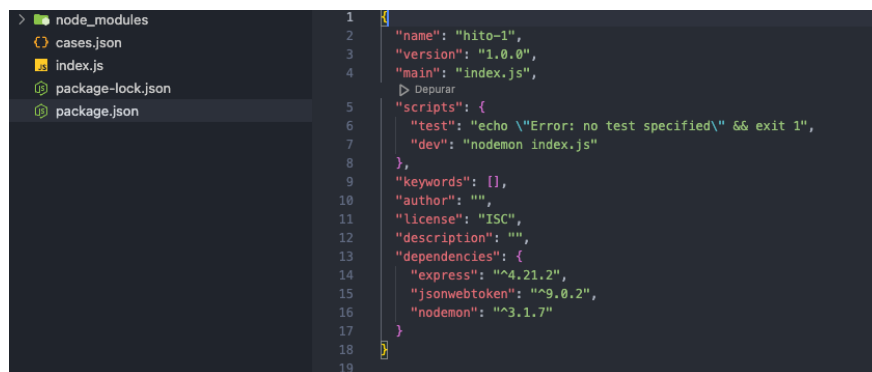
Se crea un proyecto utilizando:

npm init -y

Luego, se instalan las dependencias utilizando:

npm i express jsonwebtoken nodemon

Finalmente, así queda la estructura del proyecto junto con las dependencias ya instaladas:



The screenshot shows a code editor with two panels. The left panel displays the file structure of a project:

- node_modules
- cases.json
- index.js
- package-lock.json
- package.json

The right panel shows the content of the `package.json` file:

```
1 {
2   "name": "hito-1",
3   "version": "1.0.0",
4   "main": "index.js",
5   "scripts": {
6     "test": "echo \\Error: no test specified\\ && exit 1",
7     "dev": "nodemon index.js"
8   },
9   "keywords": [],
10  "author": "",
11  "license": "ISC",
12  "description": "",
13  "dependencies": {
14    "express": "^4.21.2",
15    "jsonwebtoken": "^9.0.2",
16    "nodemon": "^3.1.7"
17  }
18 }
```

2. **Generar al menos dos rutas, una principal y una protegida que requiera de registro y login para poder acceder a la información de dicha ruta. La información a mostrar es a libre elección.**

En este proyecto, se generará una API de una plataforma llamada “LawCases” que permite administrar los distintos casos judiciales de una empresa. En este caso se crean

- a. **GET “/”**: Ruta principal para dar la bienvenida a la API.
- b. **POST “/register”**: Ruta para registrar un usuario. Se envía a través del body un “username” y un “password” para registrarlo en la base de datos (por ahora un array).
- c. **POST “/login”**: Ruta para autenticarse y obtener un token que permitirá acceder a las distintas rutas protegidas. Se envía “username” y un “password” a través del body y en caso de estar correctos, se devuelve un token que dura 1 hora.
- d. **GET “/cases”**: Ruta para obtener un objeto con todos los casos que administra LawCases de la empresa. Es una ruta protegida, por lo que en los headers hay que enviar el token y en caso de ser valido, se devuelve el objeto.
- e. **GET “/cases/:id”**: Ruta para obtener un objeto con un caso específico del portafolio. Es una ruta protegida, por lo que en los headers hay que enviar el token y un parámetro con el

id del caso buscado. En caso de que el token sea válido y que el id existe, se devuelve un objeto con la información de dicho caso.

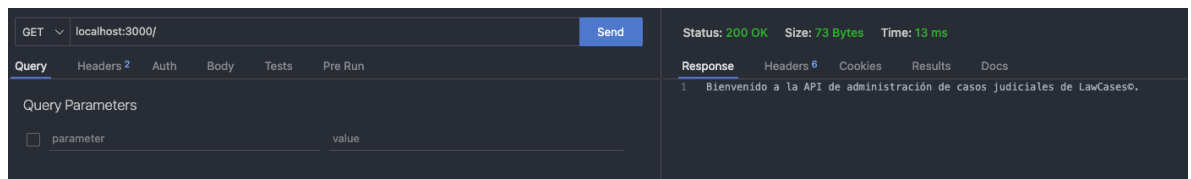
3. **Generar pruebas de funcionamiento incorporando en un archivo de texto, puede ser en Word o Google Docs. En este documento existirán pantallazos que evidencian cada una de las acciones GET y POST requeridas.**

Inicializamos el servidor con la API utilizando nodemon y el ambiente dev:

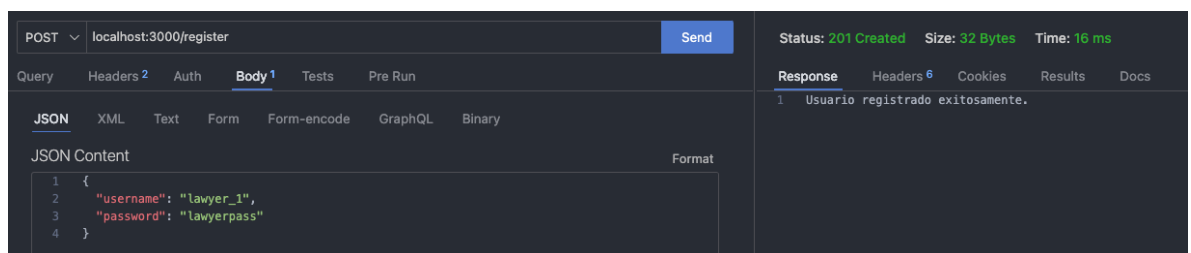
```
(base) nicolaschavez@MacBook-Pro-de-Nicolas Hito 1 % npm run dev
> hito-1@1.0.0 dev
> nodemon index.js

[nodemon] 3.1.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
Servidor encendido
```

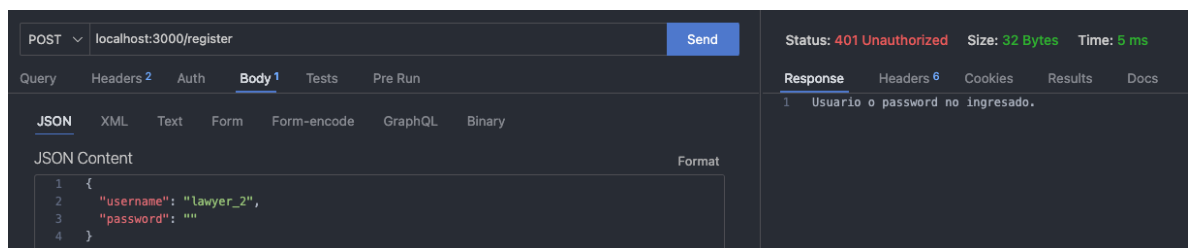
- a. **GET “/”:** Obtenemos un mensaje de bienvenida para indicarnos que el servidor y la API están activos.



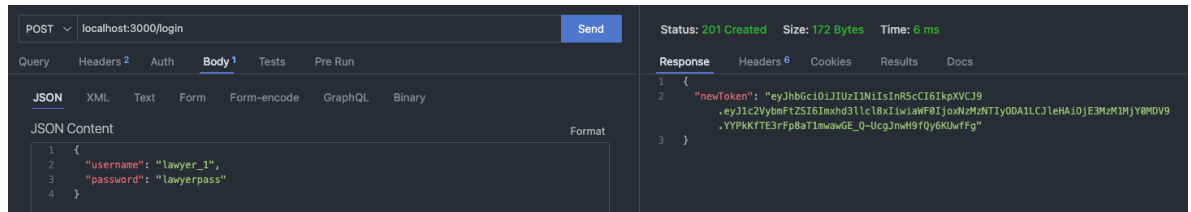
- b. **POST “/register”:** Nos registramos enviando un usuario y una contraseña en el body y recibimos un status 201 indicando que hemos realizado el registro correctamente.



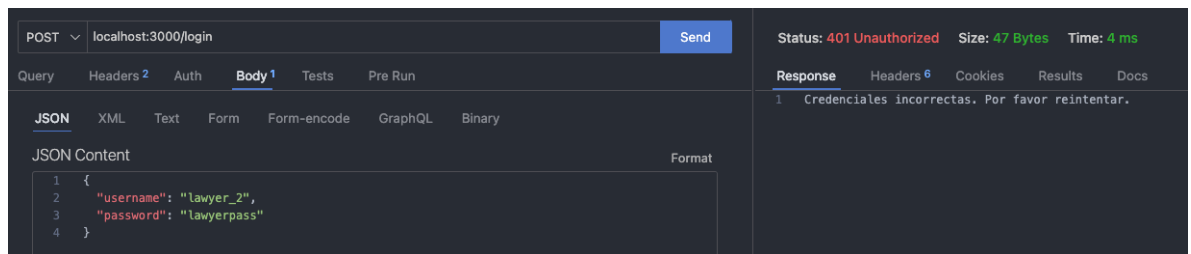
La API también maneja los casos en donde no se haya ingresado el usuario o la contraseña:



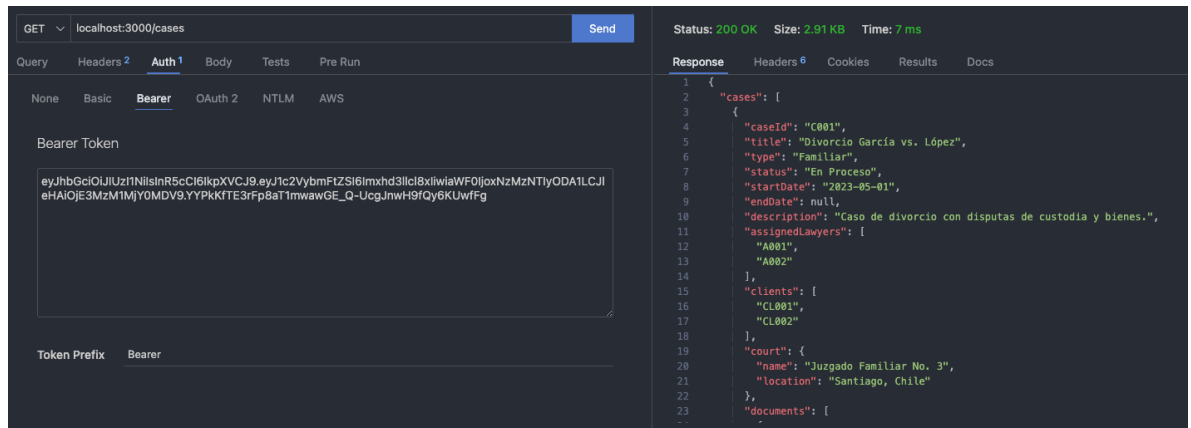
- c. **POST “/login”**: Enviamos las credenciales creadas anteriormente para recibir un token que nos permitirá autenticarnos para obtener los casos.



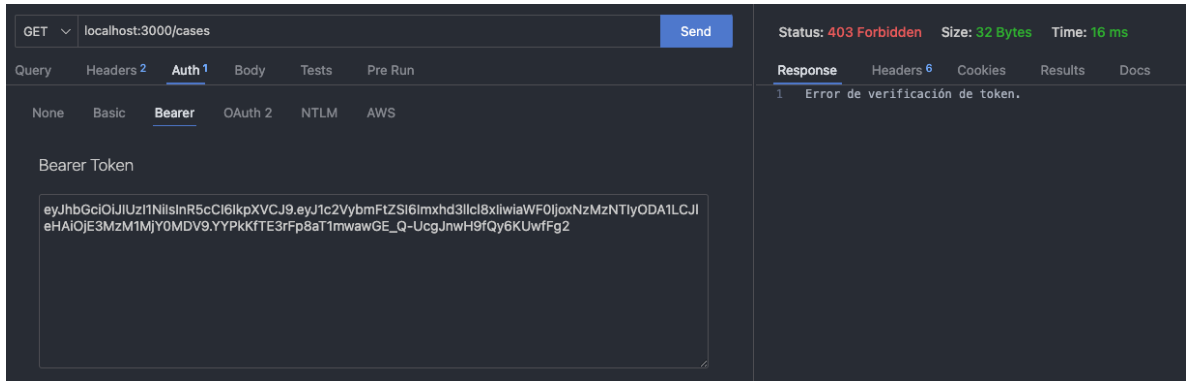
La API también maneja los casos en donde las credenciales sean incorrectas.



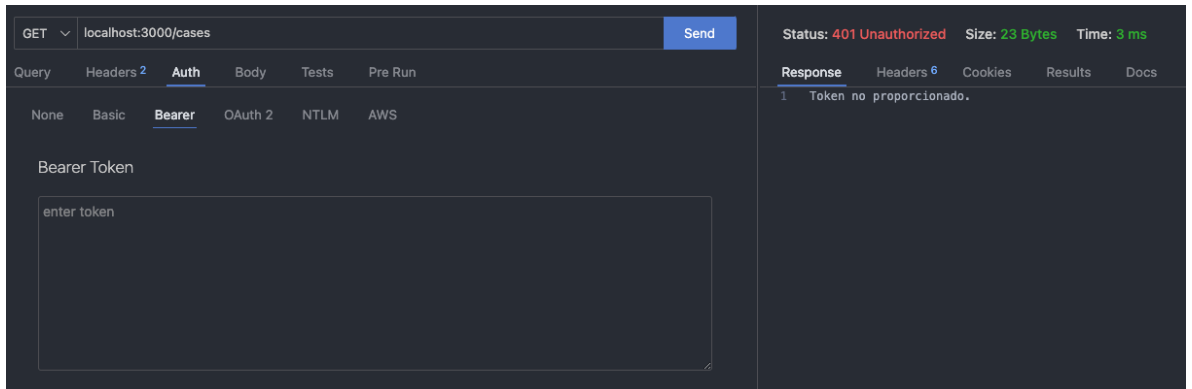
- d. **POST “/cases”**: Con el token recibido anteriormente, lo enviamos a través de “Auth” para recibir todos los casos del portafolio.



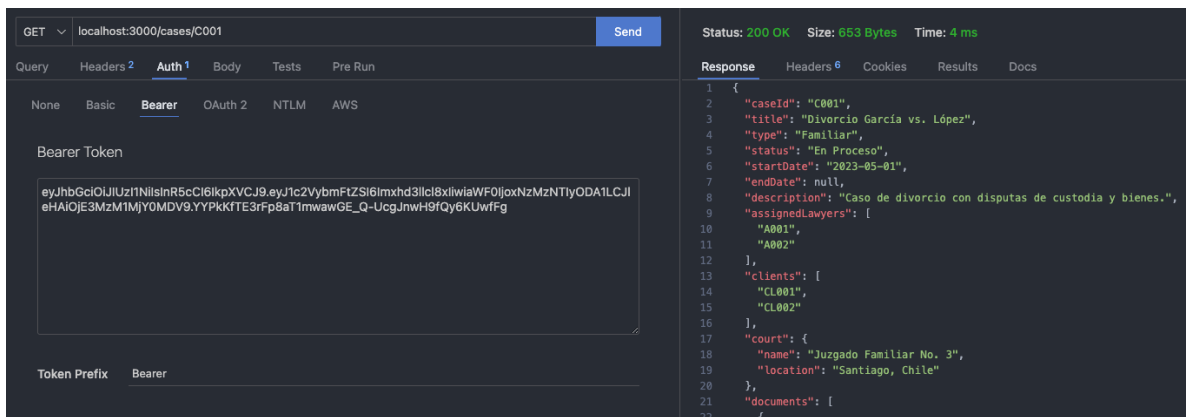
La API también maneja los casos en donde el token no es válido.



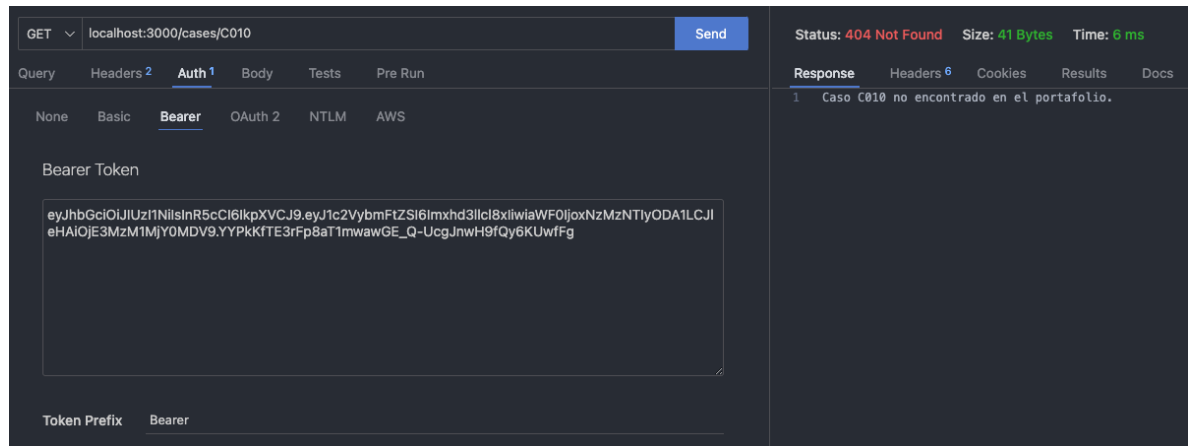
Además, también se maneja el caso cuando no se especifica el token.



- e. **POST “/cases/:id”:** Con el token recibido anteriormente, también podemos buscar un caso específico mediante el id entregado como parámetro.



También se maneja el caso cuando el id no existe.



The screenshot displays a REST client interface with the following details:

- Request:**
 - Method: GET
 - URL: localhost:3000/cases/C010
 - Auth: Bearer
 - Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Imxhd3llcl8xliwiaWF0IjoxNzY0MDV9.YYPkKfTE3rFp8aT1mwawGE_Q-UcgJnwH9fQy6KUwfFg
- Response:**
 - Status: 404 Not Found
 - Size: 41 Bytes
 - Time: 6 ms
 - Message: 1 Caso C010 no encontrado en el portafolio.