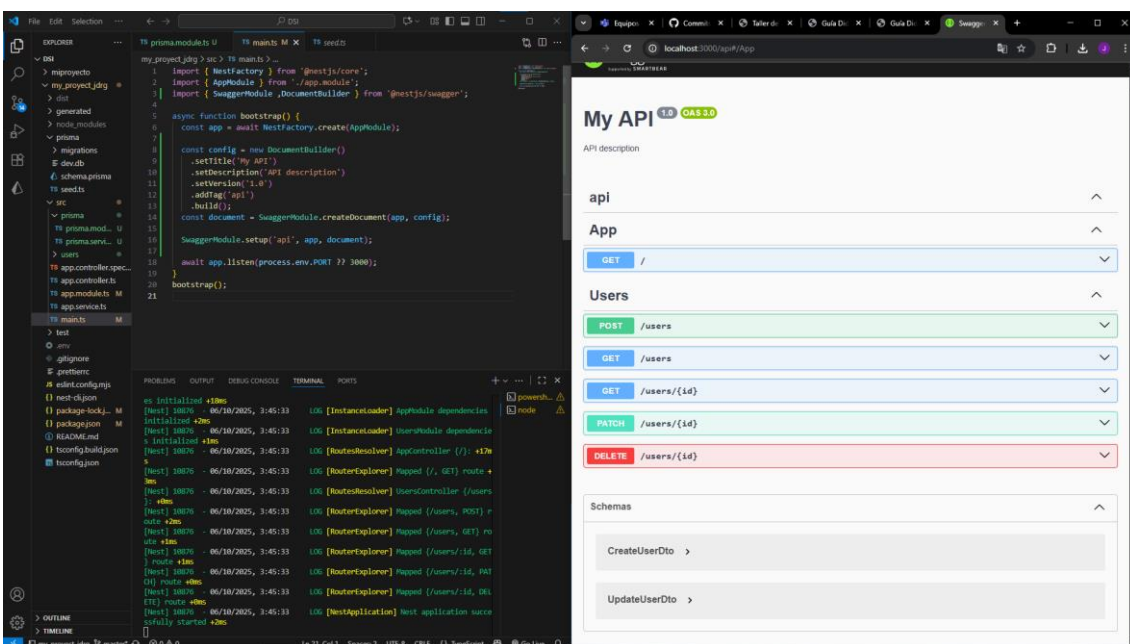
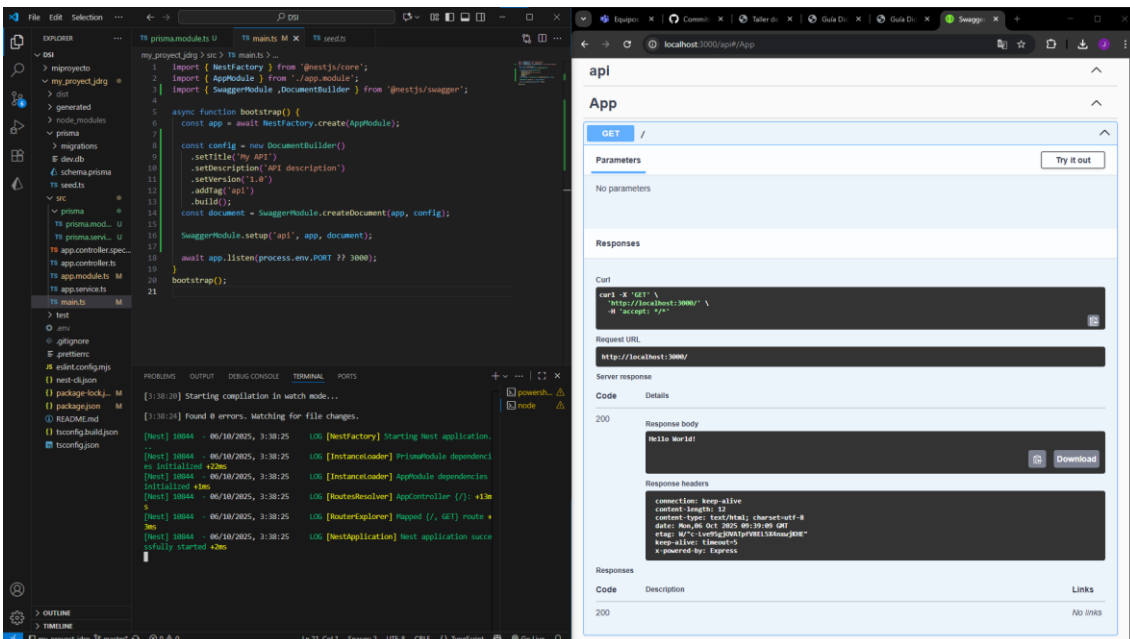


Inicio

<https://github.com/JoseRivera14/Guia-Didactica.git>



Get all users

The screenshot shows a development environment with VS Code on the left and Swagger UI on the right. In VS Code, the `userService.ts` file is open, showing the `findAll()` method which calls `this.prisma.user.findMany()`. The Swagger UI on the right displays the `findAll` endpoint with a response body containing an array of user objects. The response body is as follows:

```
{
  "id": 1,
  "email": "adam@technolutions.com",
  "name": "Adam Tech Solutions",
  "password": "5108105a02063208b3.Cz2ac0f9t5v0f.5kq6v0v0h0p0z0g0j0",
  "createdAt": "2025-10-06T09:11:07.752Z",
  "updatedAt": "2025-10-06T09:11:07.752Z",
  "role": "ADMIN",
  "tenantId": 1
},
{
  "id": 2,
  "email": "user@technolutions.com",
  "name": "John Developer",
  "password": "5108105a02063208b3.Cz2ac0f9t5v0f.5kq6v0v0h0p0z0g0j0",
  "createdAt": "2025-10-06T09:11:07.752Z",
  "updatedAt": "2025-10-06T09:11:07.752Z",
  "role": "USER",
  "tenantId": 1
},
{
  "id": 3,
  "email": "admin@martingre.com",
  "name": "Admin Marketing Pro",
  "password": "5108105a02063208b3.Cz2ac0f9t5v0f.5kq6v0v0h0p0z0g0j0",
  "createdAt": "2025-10-06T09:11:07.752Z",
  "updatedAt": "2025-10-06T09:11:07.752Z",
  "role": "ADMIN",
  "tenantId": 1
}
```

Get find by id 2

The screenshot shows a development environment with VS Code on the left and Swagger UI on the right. In VS Code, the `userService.ts` file is open, showing the `findOne(id: number)` method which calls `this.prisma.user.findUnique({where: {id}})`. The Swagger UI on the right displays the `findOne` endpoint with a response body containing a single user object. The response body is as follows:

```
{
  "id": 2,
  "email": "user@technolutions.com",
  "name": "John Developer",
  "password": "5108105a02063208b3.Cz2ac0f9t5v0f.5kq6v0v0h0p0z0g0j0",
  "createdAt": "2025-10-06T09:11:07.752Z",
  "updatedAt": "2025-10-06T09:11:07.752Z",
  "role": "USER",
  "tenantId": 1
}
```

Create user

The screenshot shows a development environment with VS Code on the left and Swagger UI on the right. In VS Code, the file `my_project_jdk > src > users > dto > create-user.dto.ts` is open, displaying the following TypeScript code:

```
1 import { ApiProperty } from '@nestjs/swagger';
2
3 export class CreateUserDto {
4   @ApiProperty({required: true,example: 'usuario@empresa.com'})
5   email: string;
6   @ApiProperty({required: true,example: 'John Doe'})
7   name: string;
8   username: string;
9   @ApiProperty({required: true,example: 'password123'})
10  password: string;
11   @ApiProperty({required: true,example: 1,description: 'ID del tenant'})
12  tenantId: number;
13 }
14
15
16
17
18
```

The Swagger UI on the right displays the API for the `Users` controller. The `POST /users` endpoint is highlighted. It shows no parameters and a required request body of type `application/json`. The example value is:

```
{
  "email": "usuario@empresa.com",
  "name": "John Doe",
  "password": "password123",
  "tenantId": 1
}
```

The terminal at the bottom of VS Code shows the application starting successfully.

Update user

The screenshot shows the Swagger UI for the `PATCH /users/{id}` endpoint. The `Parameters` section shows a required path parameter `id` of type `string`. The `Request body` is required and of type `application/json`. The `Example Value` is:

```
{
  "email": "usuario@empresa.com",
  "name": "John Doe",
  "password": "password123",
  "tenantId": 1
}
```

The `Responses` section shows a `200` status code with no links.

Delete User

my_project_jdg > src > users > dto > create-user.dto.ts > ...
1 import { ApiProperty } from '@nestjs/swagger';
2
3 export class CreateUserDto {
4 @ApiProperty({required: true,example: 'usuario@empresa.com'})
5 email: string;
6 @ApiProperty({required: true,example: 'John Doe'})
7 name: string;
8 username?: string;
9 password: string;
10
11 @ApiProperty({required: true,example: 'password123'})
12 password2: string;
13
14 @ApiProperty({required: true,example: 1,description: 'ID del tenant'})
15 tenantId: number;
16 }
17
18

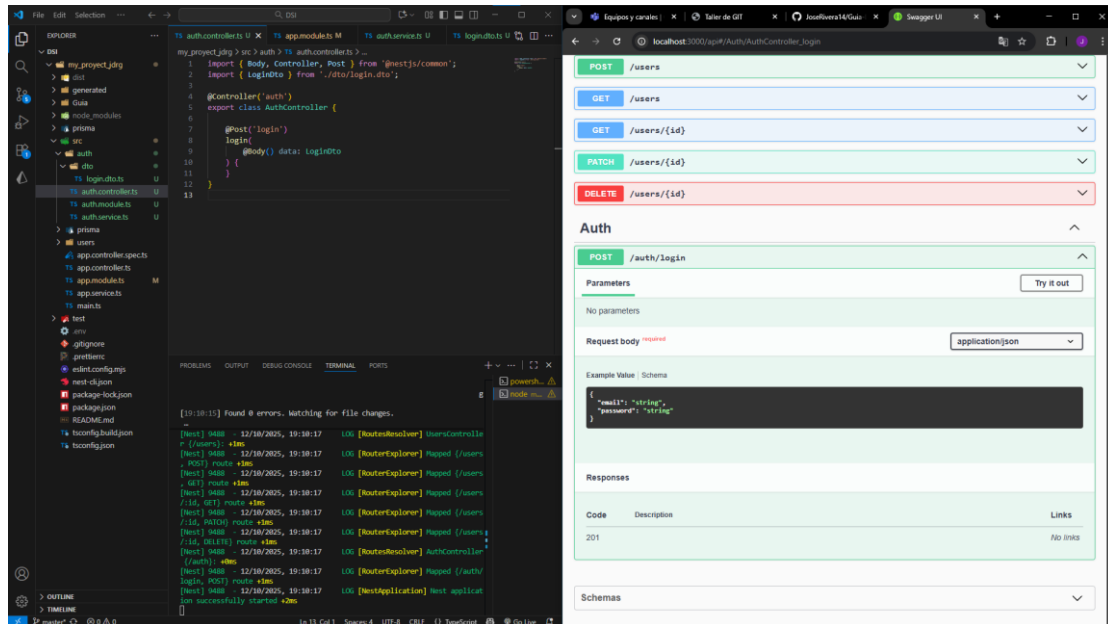
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [InstanceLoader] AppModule dependencies initialized +2ms
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [InstanceLoader] UsersModule dependencies initialized +3ms
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [RoutesResolver] AppController (/): +17ms
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [RouterExplorer] Mapped {/, GET} route +5ms
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [RouterExplorer] Mapped {/users, GET} route +4ms
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [RouterExplorer] Mapped {/users, POST} route +4ms
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [RouterExplorer] Mapped {/users/:id, GET} route +1ms
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [RouterExplorer] Mapped {/users/:id, PUT} route +1ms
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [RouterExplorer] Mapped {/users/:id, DELETE} route +1ms
[Nest] 13076 - 06/10/2025, 4:15:31 LOG [NestApplication] Nest application successfully started +2ms

Responses
Code Description Links
200 No links

DELETE /users/{id} Try it out
Parameters
Name Description
id *required string (path)
Responses
Code Description Links
200 No links

Schemas
CreateUserDto >
UpdateUserDto >

Auth



Este error lo revisé por 2h pero no encontré la solución (error 401 en el auth)

Probe con todos los correos del seed

Execute

Clear

Responses

Curl

```
curl -X 'POST' \
'http://localhost:3000/auth/login' \
-H 'accept: */*' \
-H 'Content-Type: application/json' \
-d '{
  "email": "user@consultingexperts.com",
  "password": "$2b$10$qdu2M6832WKBxJ.CxZZmz0FqMSzoY.9AfQmUdYvW8zhpmu2tgi/ji"
}'
```

Request URL

```
http://localhost:3000/auth/login
```

Server response

Code	Details
401	Error: Unauthorized

Undocumented

Response body

```
{
  "message": "Credenciales inválidas",
  "error": "Unauthorized",
  "statusCode": 401
}
```

Download

Response headers

```
connection: keep-alive
content-length: 77
content-type: application/json; charset=utf-8
date: Mon, 13 Oct 2025 03:48:59 GMT
etag: W/"4d-Wtkv+T/44N4290jcIjb0Fr2crac"
keep-alive: timeout=5
x-powered-by: Express
```

Responses

