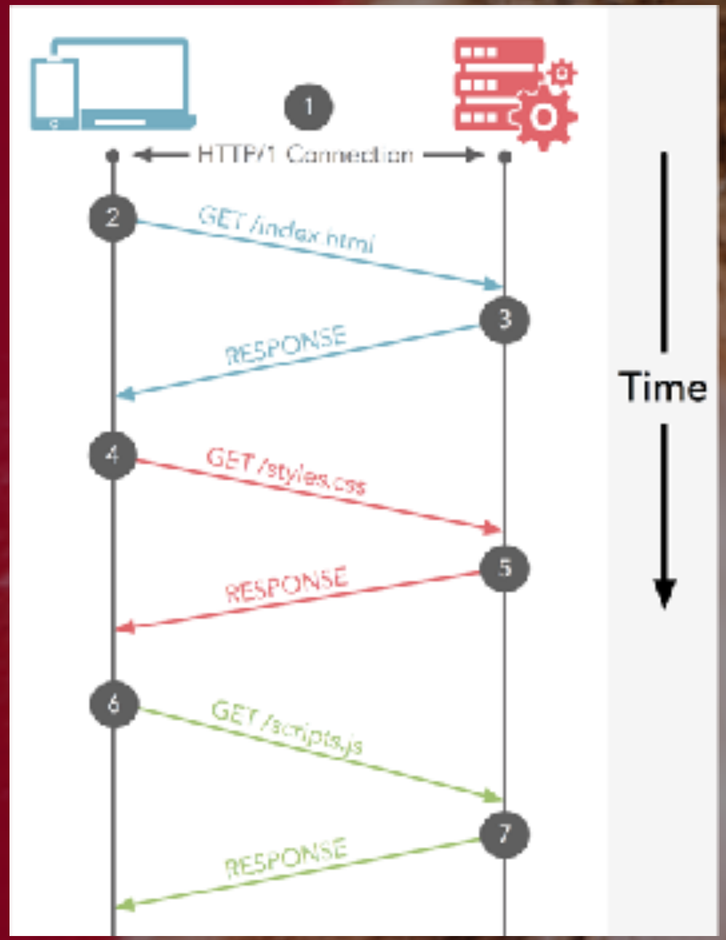
MIGUEL CAMPOS RIVERA
@miguelcampos

NESTJS FRAMEWORK: INTRODUCCIÓN

GDG SEVILLA
ANDALUCIA OPEN FUTURE
Miércoles 12 DICIEMBRE 2018

nest



1 INSTALACIÓN
npm

2 CONTROLLERS
HTTP Request

3 PROVIDERS
TypeORM, DB

4 EXTRAS
Swagger, Middlewares

5 DEPLOY
Run & deploy

1 INTRO

nest

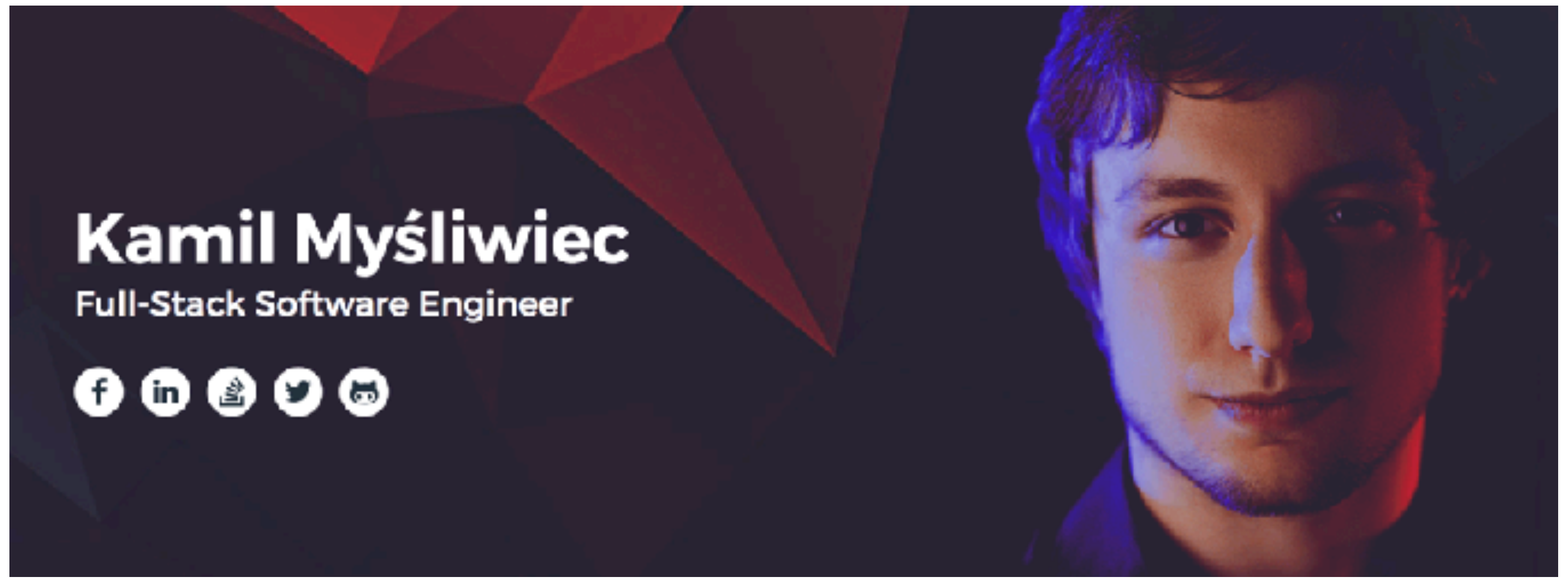
¿QUÉ ES NESTJS?

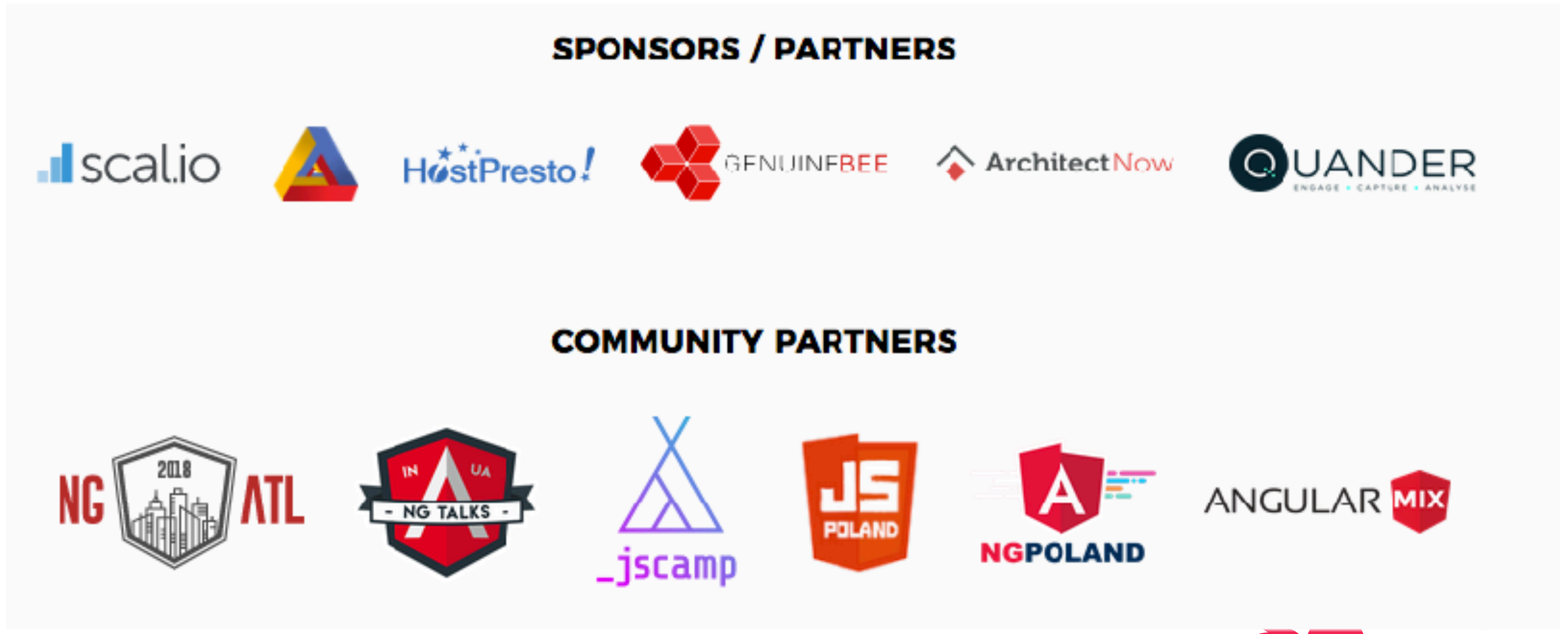
nestjs.com

A progressive **Node.js** framework for building efficient, reliable and scalable server-side applications.

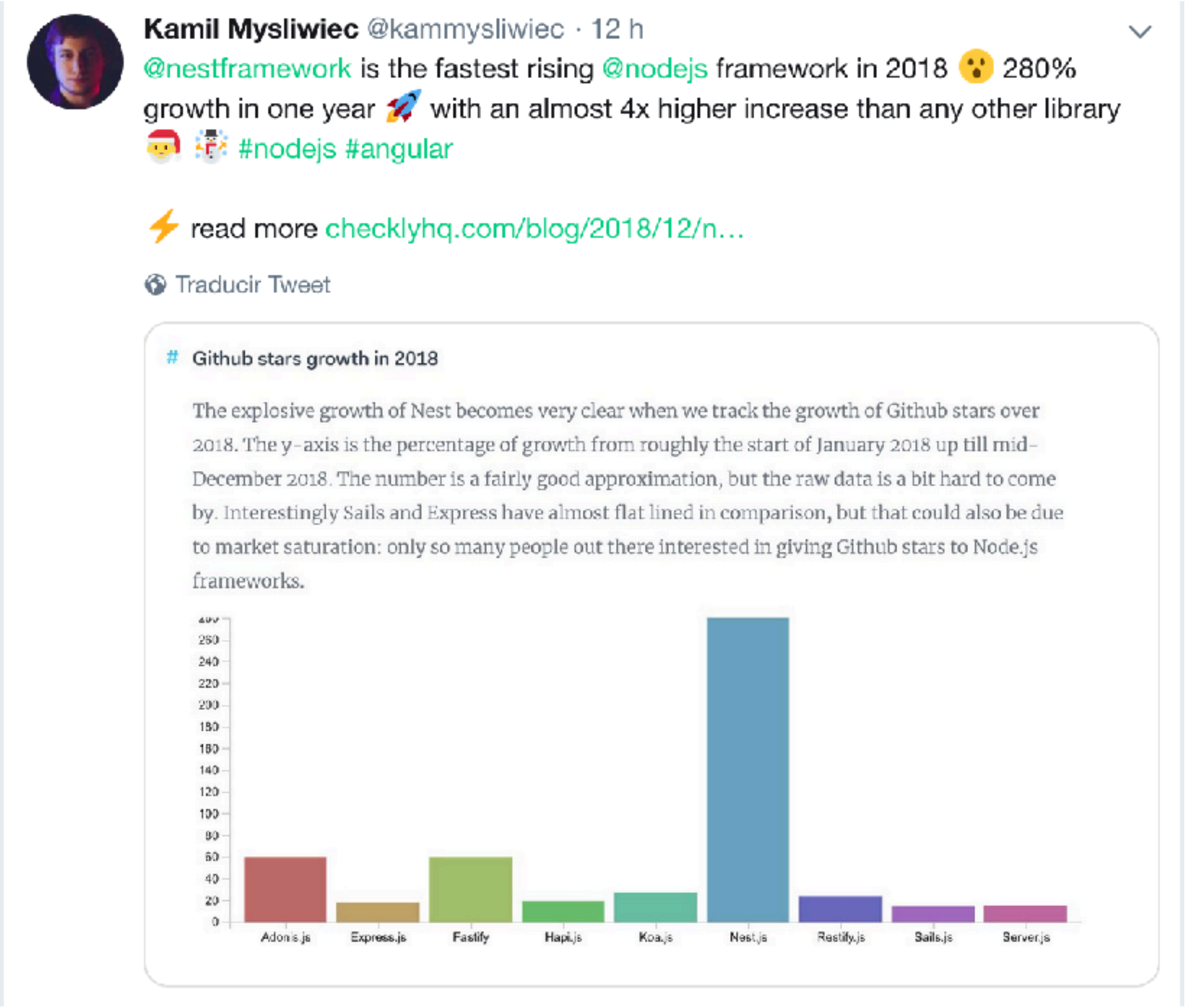
1 INTRO

¿QUIÉN ESTÁ DETRÁS?




SPONSORS / PARTNERS

COMMUNITY PARTNERS


Kamil Mysliwiec @kammysliwiec · 12 h

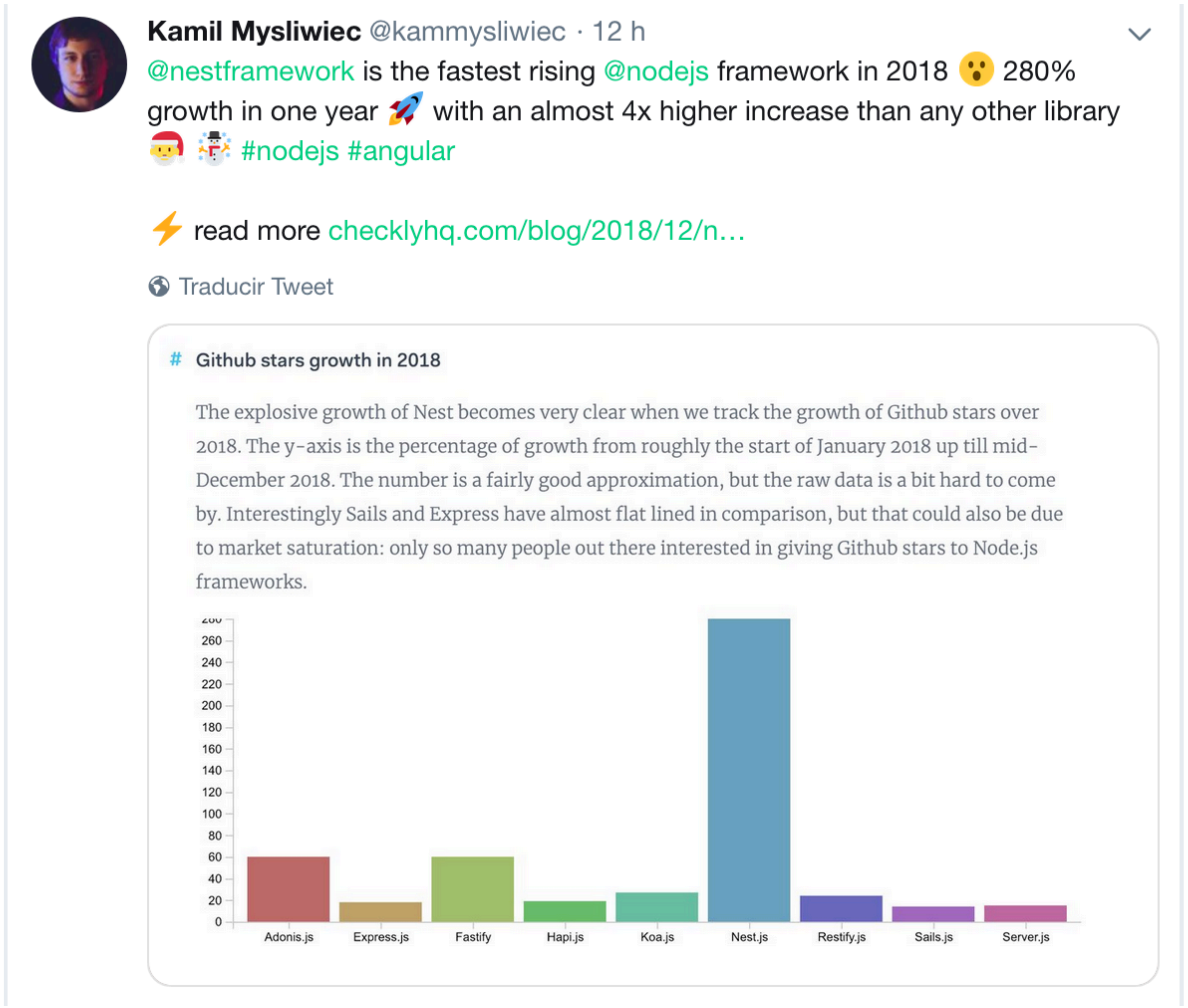
@nestframework is the fastest rising @nodejs framework in 2018 😮 280% growth in one year 🚀 with an almost 4x higher increase than any other library 🎅 ⛄ #nodejs #angular

⚡ read more checklyhq.com/blog/2018/12/n...

Traducir Tweet

Github stars growth in 2018

The explosive growth of Nest becomes very clear when we track the growth of Github stars over 2018. The y-axis is the percentage of growth from roughly the start of January 2018 up till mid-December 2018. The number is a fairly good approximation, but the raw data is a bit hard to come by. Interestingly Sails and Express have almost flat lined in comparison, but that could also be due to market saturation: only so many people out there interested in giving Github stars to Node.js frameworks.



Kamil Mysliwiec @kammysliwiec · 12 h

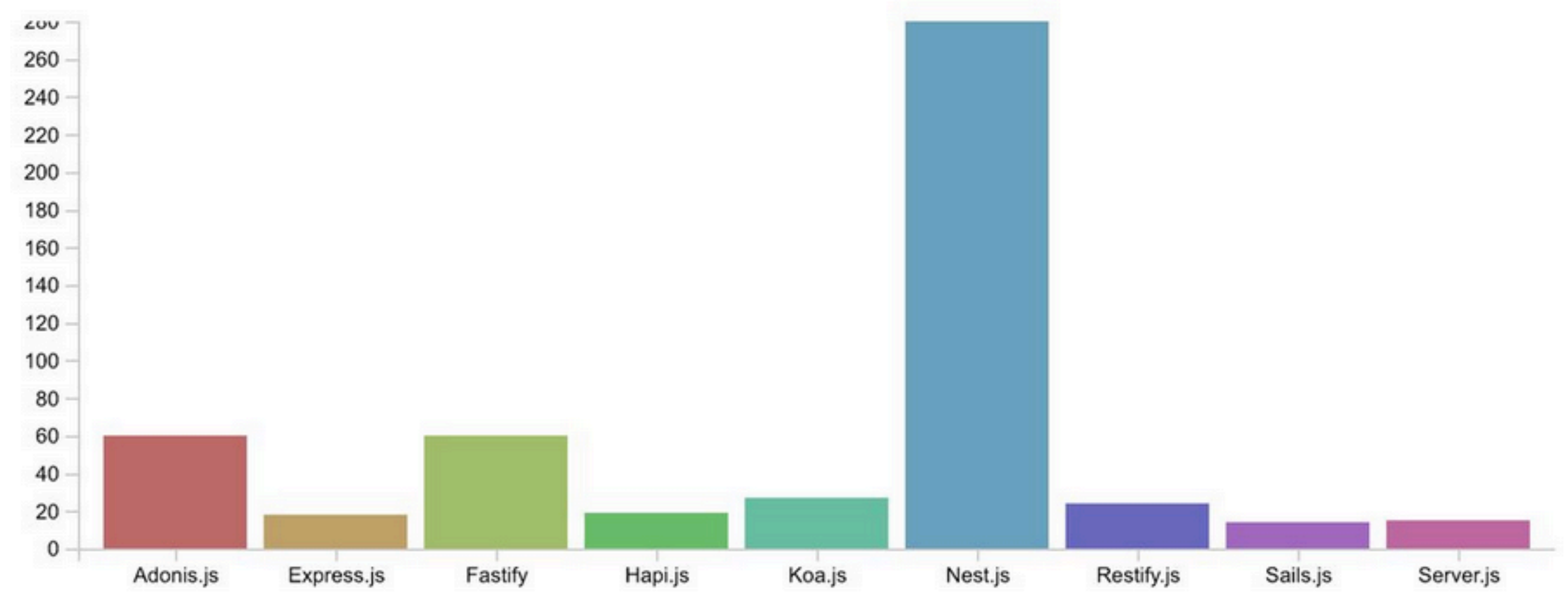
@nestframework is the fastest rising @nodejs framework in 2018 😮 280% growth in one year 🚀 with an almost 4x higher increase than any other library 🎅 ☃️ #nodejs #angular

⚡ read more checklyhq.com/blog/2018/12/n…

Traducir Tweet

Github stars growth in 2018

The explosive growth of Nest becomes very clear when we track the growth of Github stars over 2018. The y-axis is the percentage of growth from roughly the start of January 2018 up till mid-December 2018. The number is a fairly good approximation, but the raw data is a bit hard to come by. Interestingly Sails and Express have almost flat lined in comparison, but that could also be due to market saturation: only so many people out there interested in giving Github stars to Node.js frameworks.

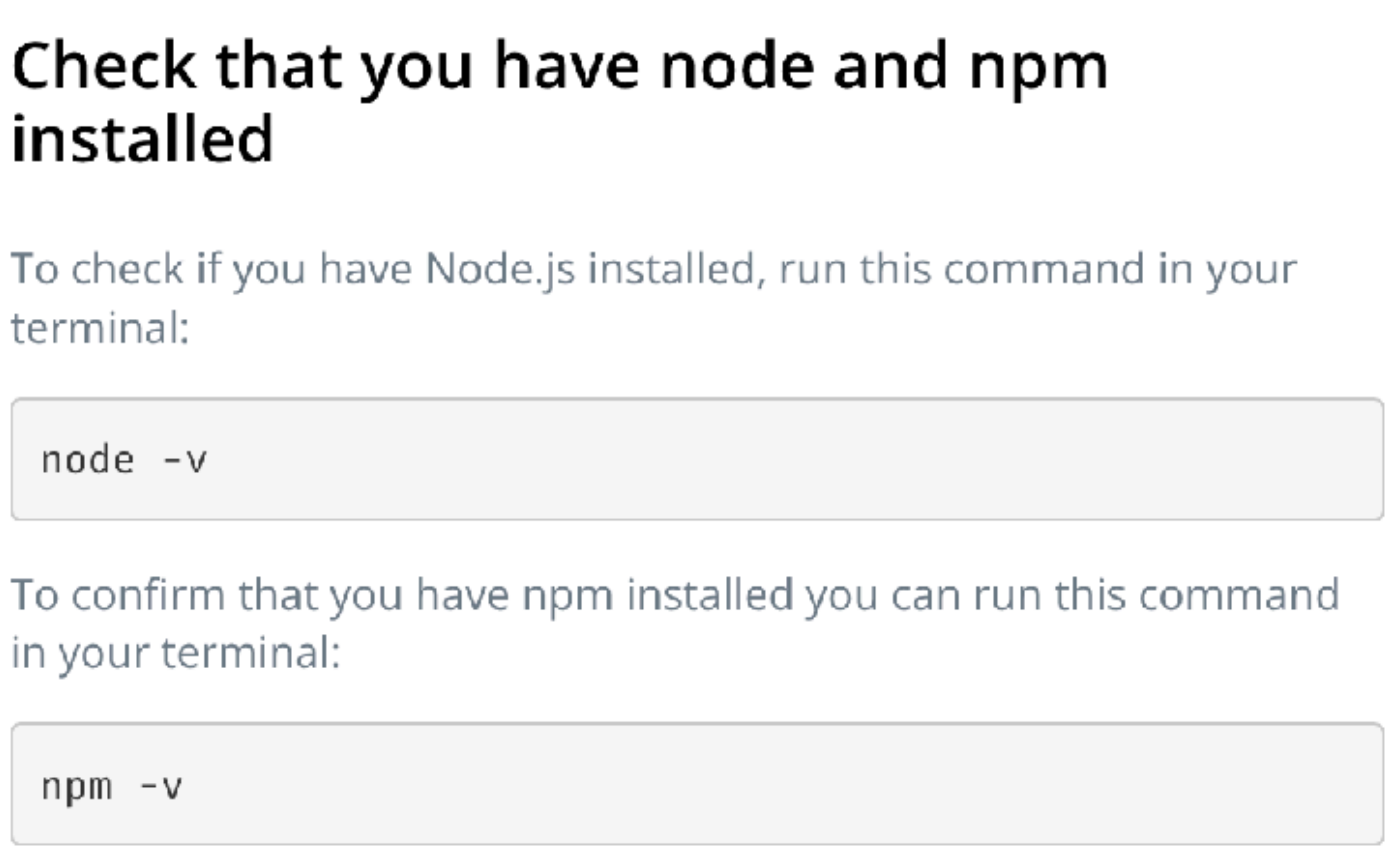


nest

1 INSTALACIÓN

nest

PREREQUISITOS
Node.js (>= 8.9.0)



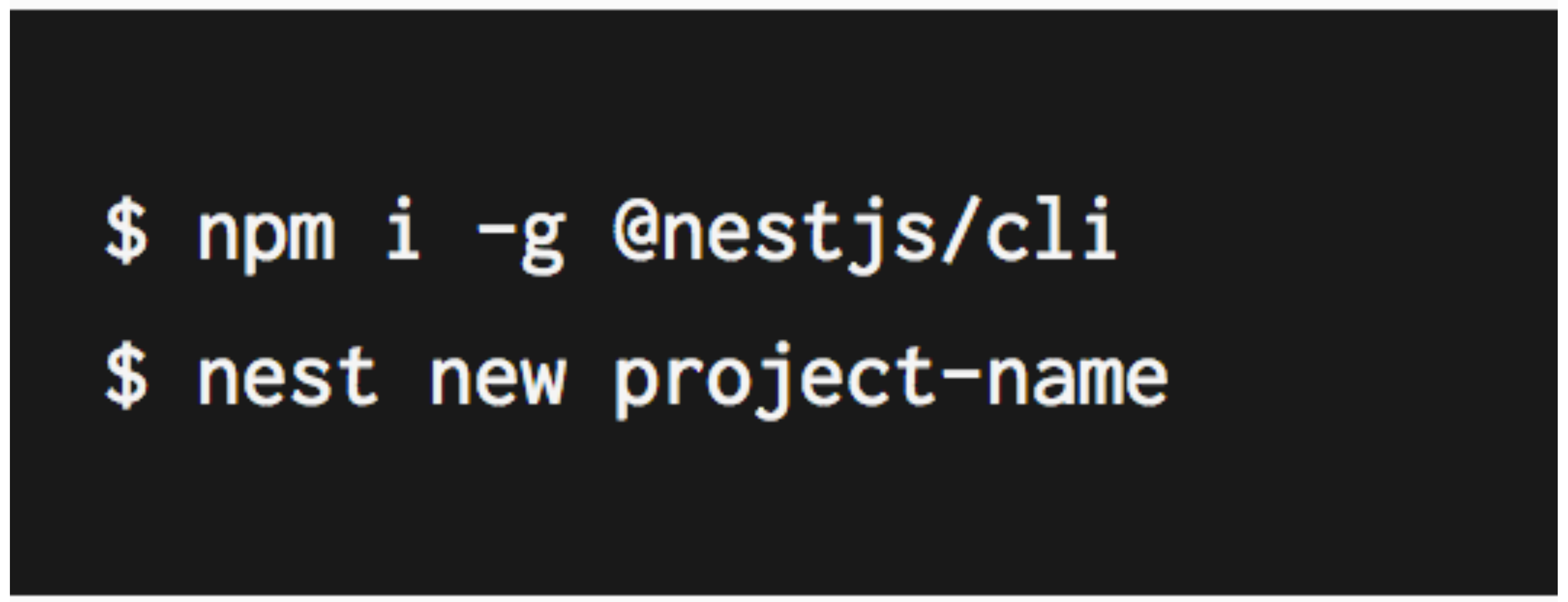
Check that you have node and npm installed

To check if you have Node.js installed, run this command in your terminal:

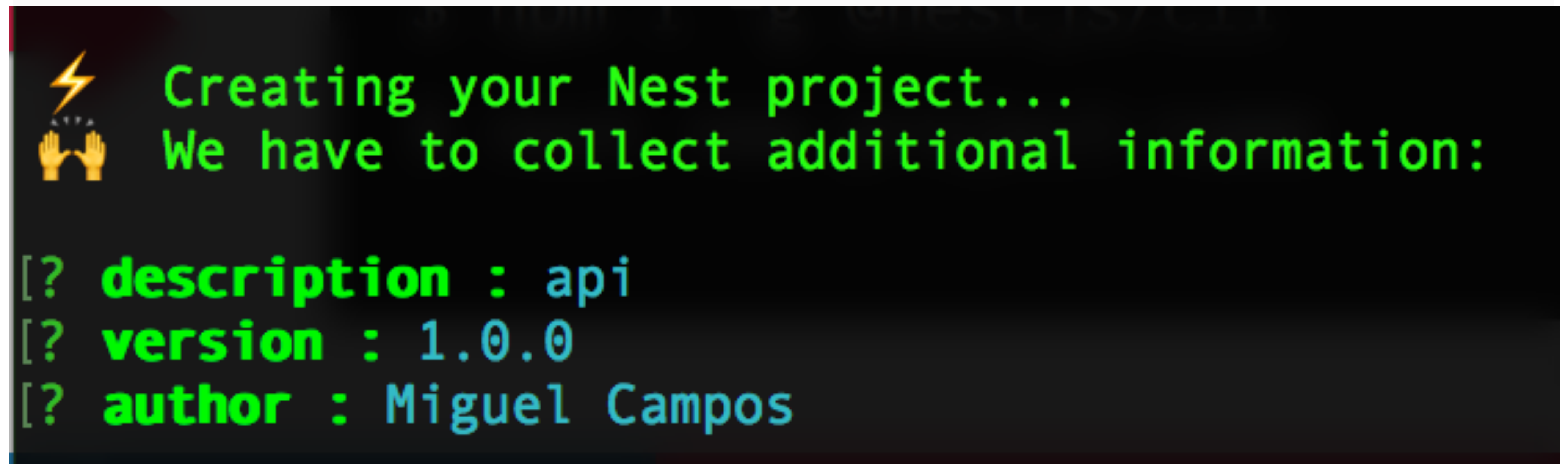
```
node -v
```

To confirm that you have npm installed you can run this command in your terminal:

```
npm -v
```



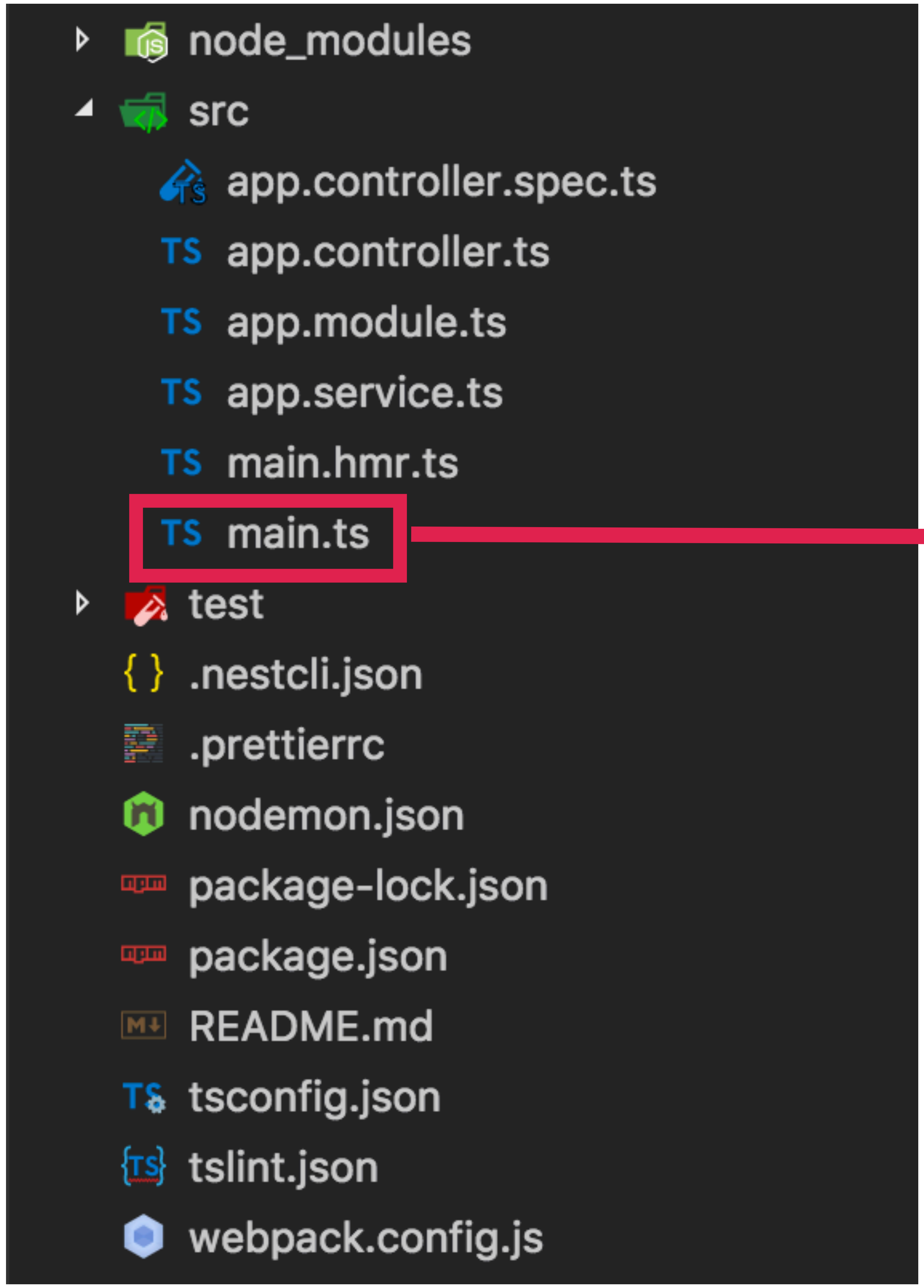
```
$ npm i -g @nestjs/cli
$ nest new project-name
```



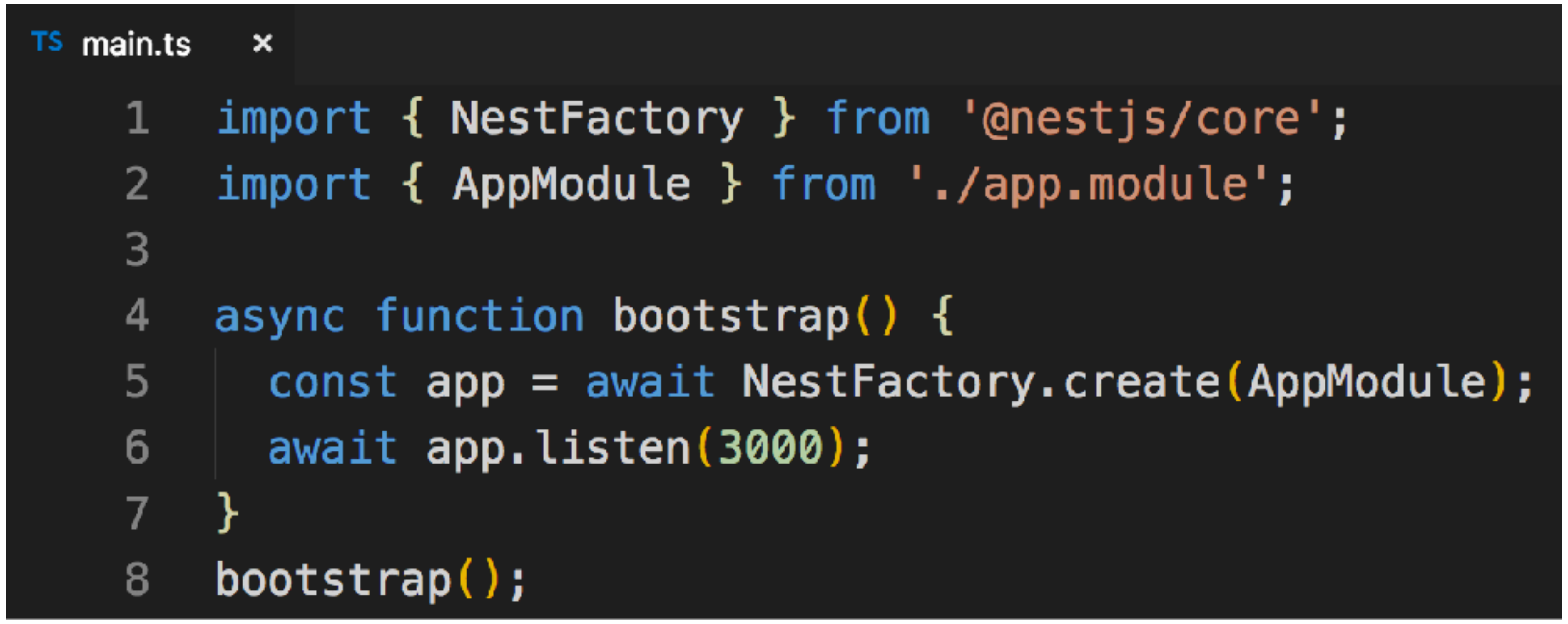
```
⚡  Creating your Nest project...
🙌  We have to collect additional information:

[? description : api
[? version : 1.0.0
[? author : Miguel Campos
```

miguelcampos

ESTRUCTURA PROYECTO NESTJS



```
▸ node_modules
▾ src
    app.controller.spec.ts
    app.controller.ts
    app.module.ts
    app.service.ts
    main.hmr.ts
    main.ts
▸ test
  .nestcli.json
  .prettierrc
  nodemon.json
  package-lock.json
  package.json
  README.md
  tsconfig.json
  tslint.json
  webpack.config.js
```



main.ts

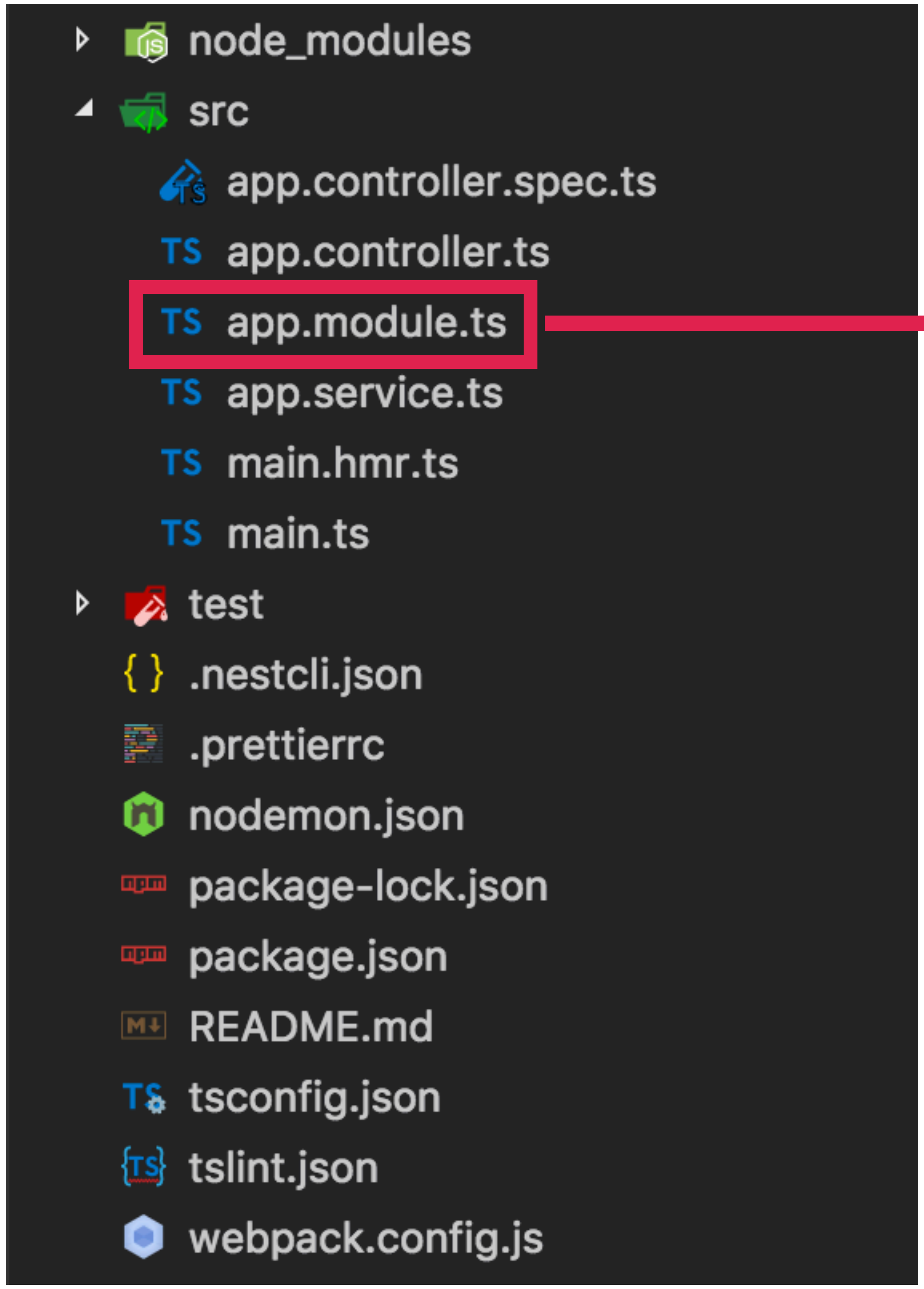
```ts
import { NestFactory } from '@nestjs/core';
import { AppModule } from './app.module';

async function bootstrap() {
  const app = await NestFactory.create(AppModule);
  await app.listen(3000);
}
bootstrap();
```

nest

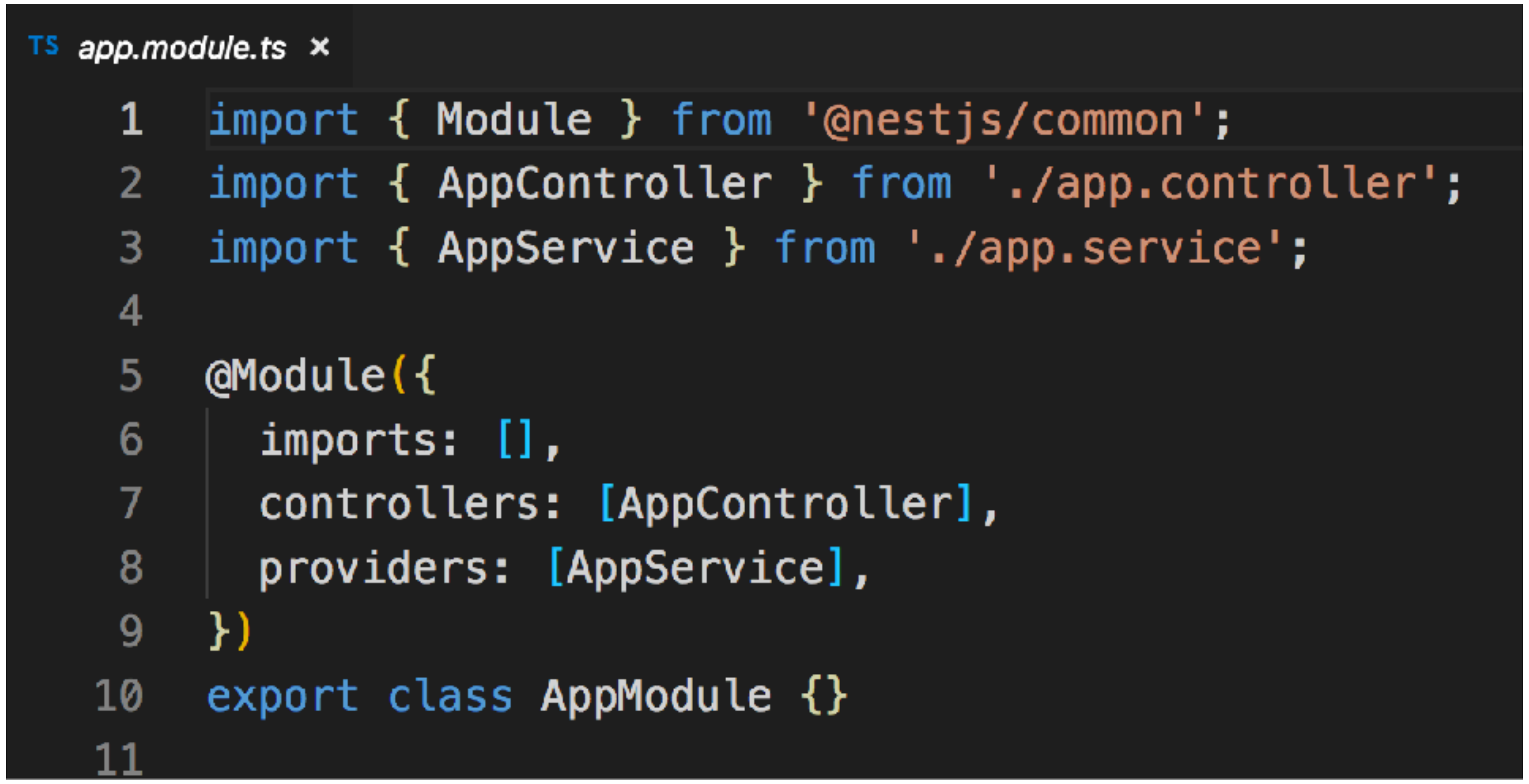
miguelcampos

ESTRUCTURA PROYECTO NESTJS

nest



```
▸ node_modules
▾ src
    app.controller.spec.ts
    app.controller.ts
    app.module.ts
    app.service.ts
    main.hmr.ts
    main.ts
▸ test
  .nestcli.json
  .prettierrc
  nodemon.json
  package-lock.json
  package.json
  README.md
  tsconfig.json
  tslint.json
  webpack.config.js
```



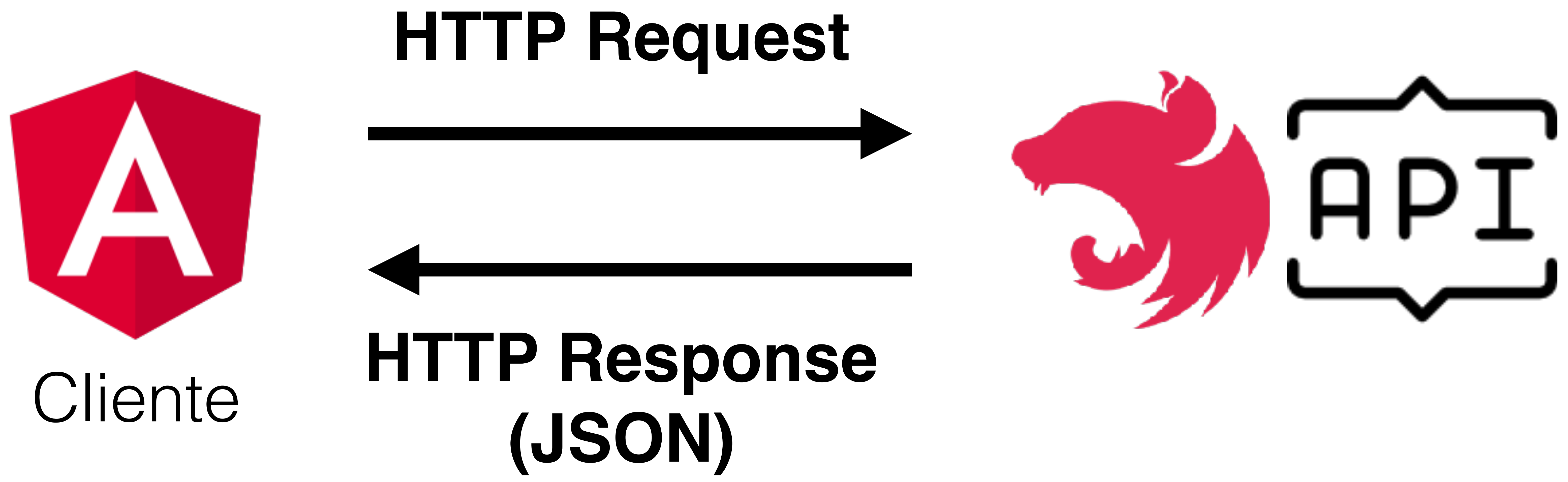
app.module.ts

```
import { Module } from '@nestjs/common';
import { AppController } from './app.controller';
import { AppService } from './app.service';

@Module({
  imports: [],
  controllers: [AppController],
  providers: [AppService],
})
export class AppModule {}

```

miguelcampos

2 CONTROLLERS

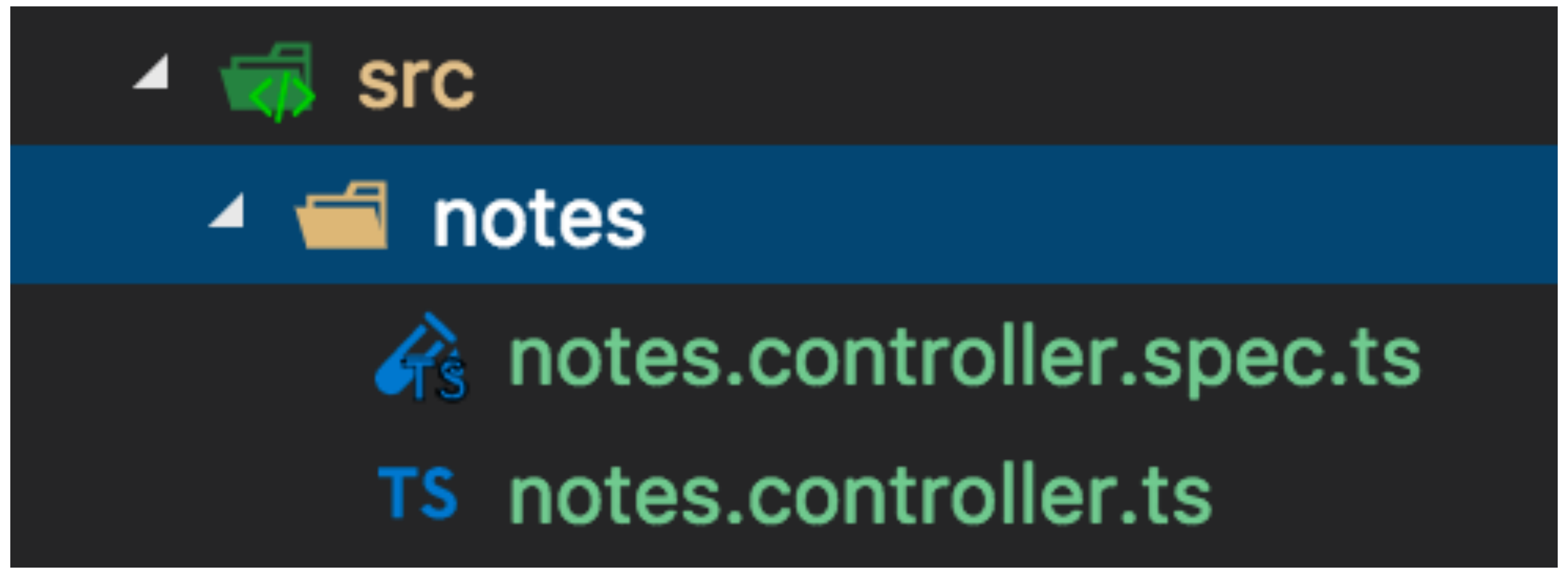
nest



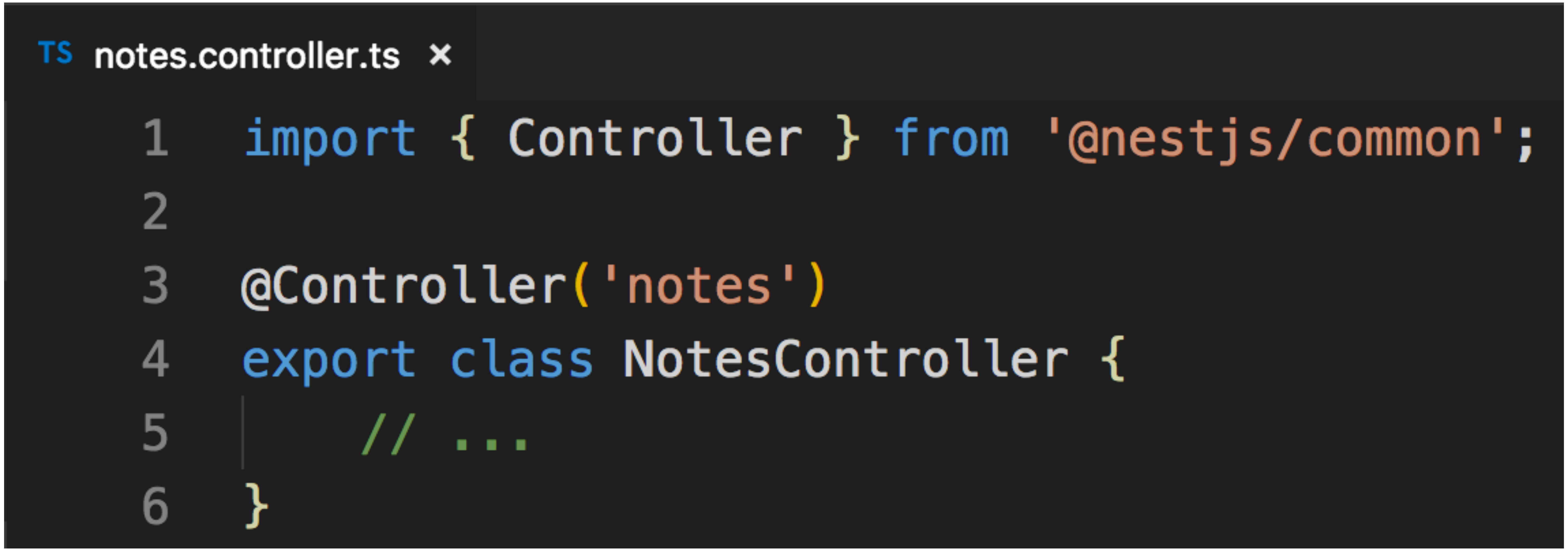
2 CONTROLLERS

nest

```
$ nest g co notes
```



```
src
  notes
    notes.controller.spec.ts
    notes.controller.ts
```



notes.controller.ts

```typescript
import { Controller } from '@nestjs/common';

@Controller('notes')
export class NotesController {
    // ...
}
```

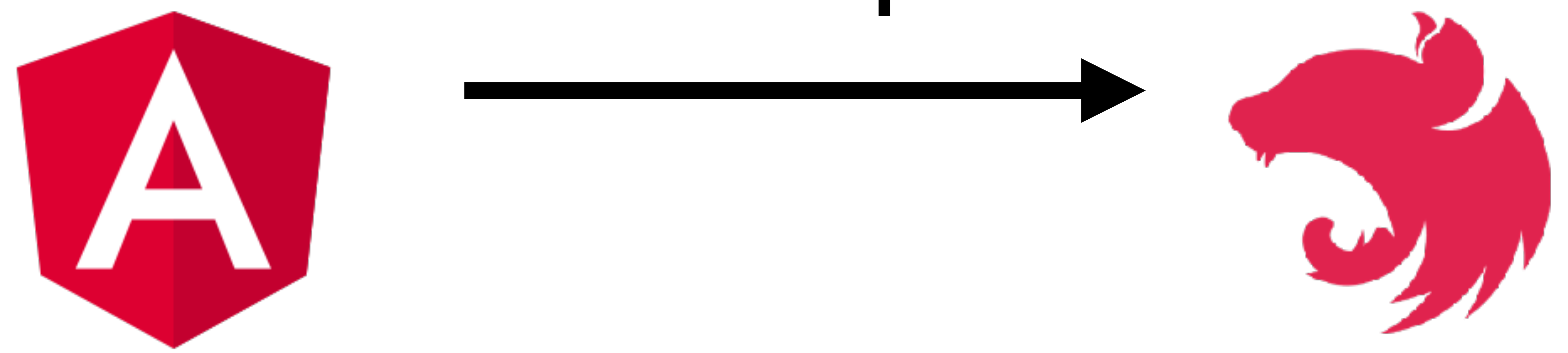
miguelcampos

2 CONTROLLERS

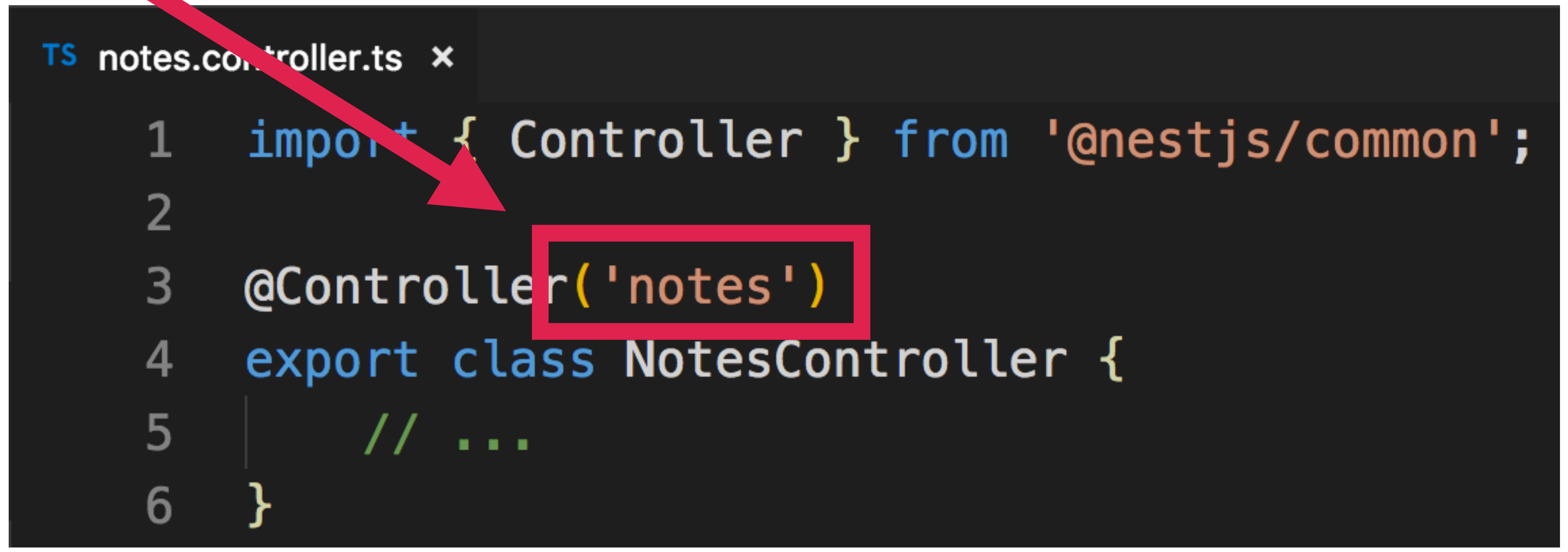
http://www.myapp.com/notes

ROUTING

HTTP Request



Cliente



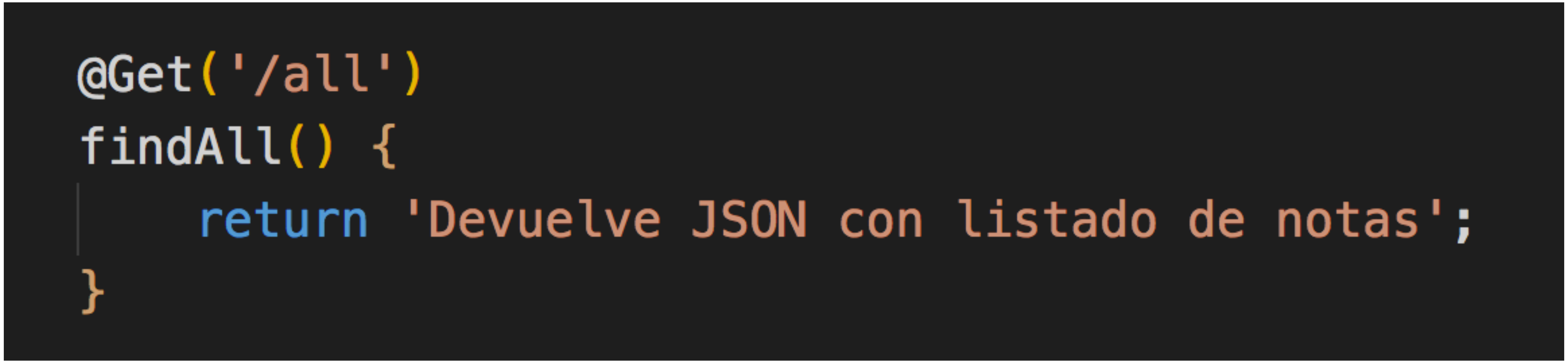
notes.controller.ts

```
import { Controller } from '@nestjs/common';

@Controller('notes')
export class NotesController {
    // ...
}
```

CONTROLLERS

nest

GET http://www.myapp.com/notes/**all**

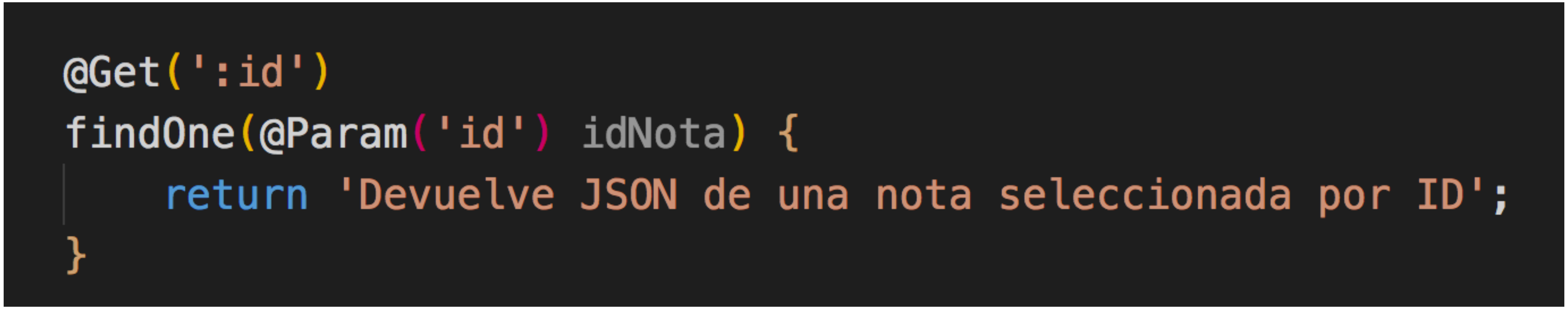


```
@Get('/all')
findAll() {
    return 'Devuelve JSON con listado de notas';
}
```

2 CONTROLLERS

nest

GET http://www.myapp.com/notes/1

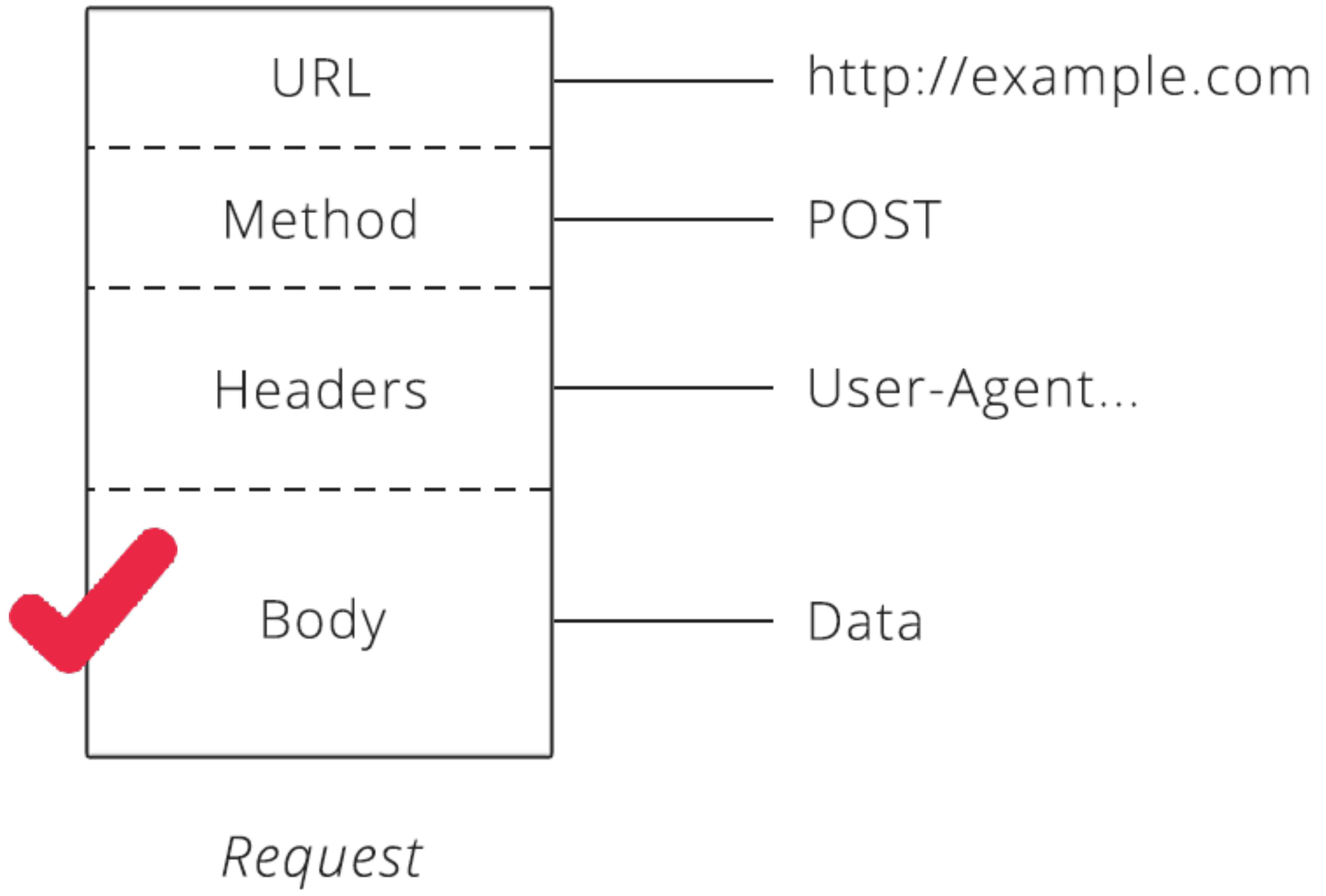


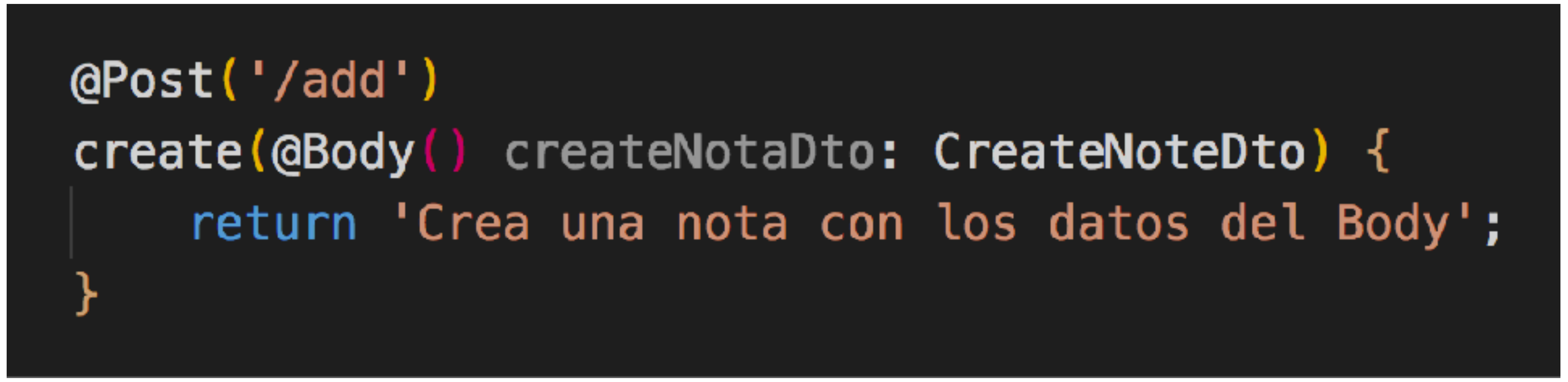
```
@Get(':id')
findOne(@Param('id') idNota) {
    return 'Devuelve JSON de una nota seleccionada por ID';
}
```

2 CONTROLLERS

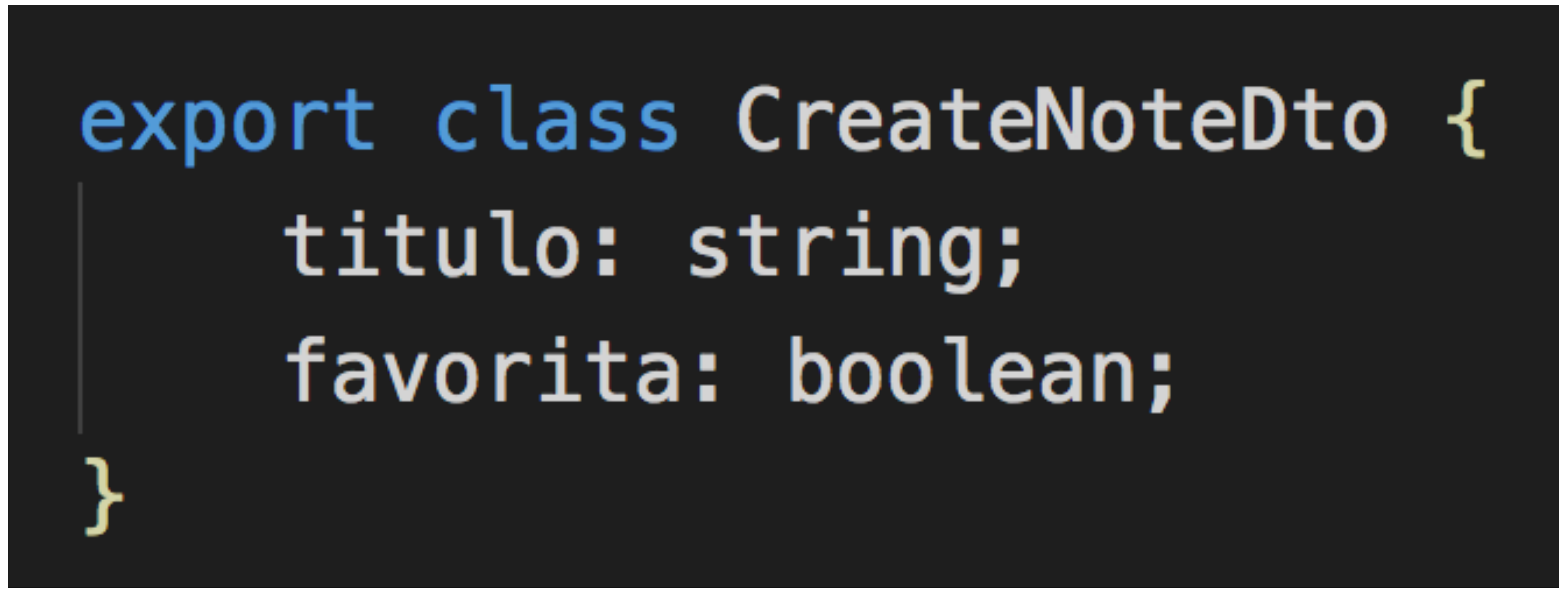
nest

POST http://www.myapp.com/notes/**add**





```
@Post('/add')
create(@Body() createNotaDto: CreateNoteDto) {
    return 'Crea una nota con los datos del Body';
}
```



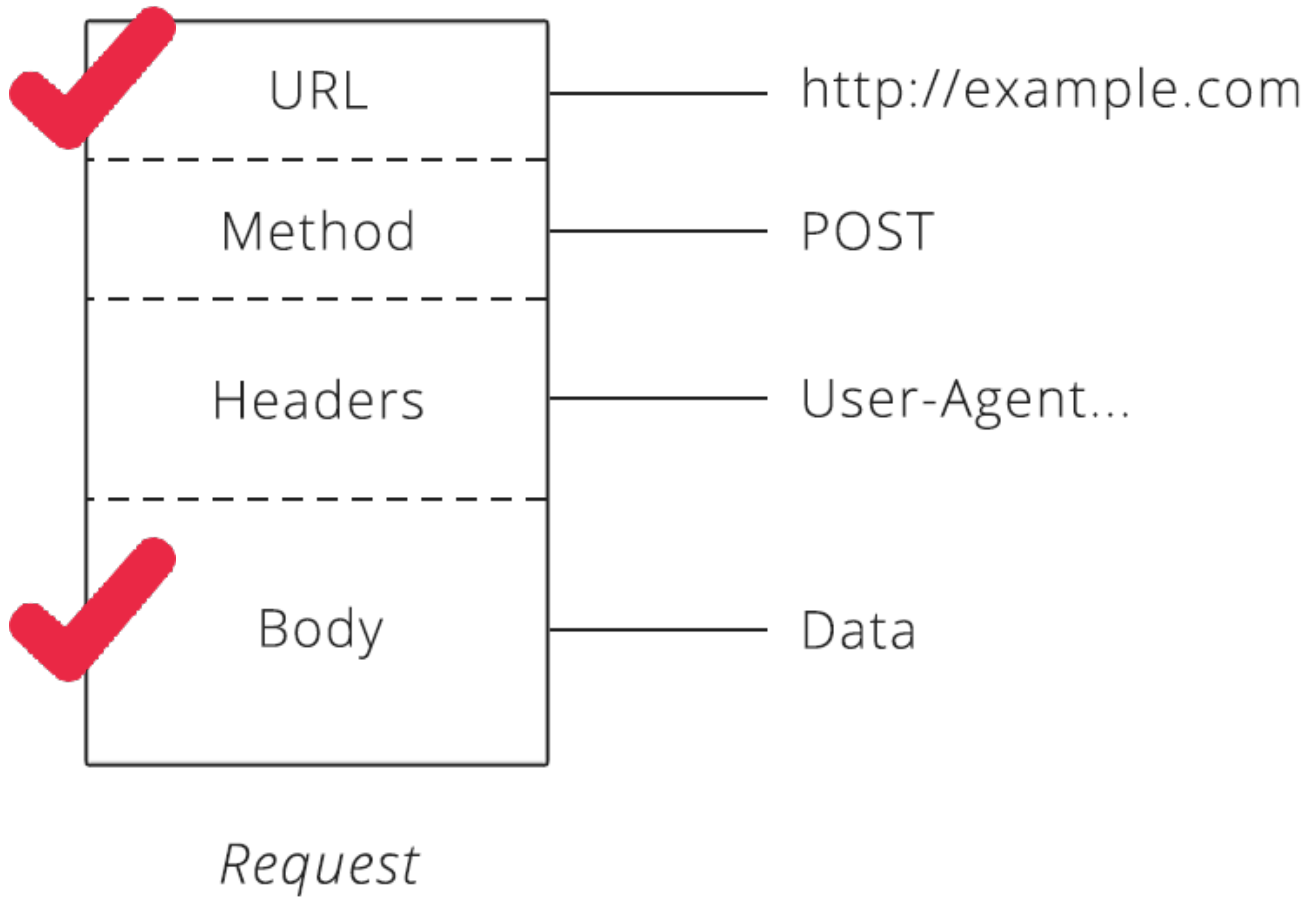
```
export class CreateNoteDto {
    titulo: string;
    favorita: boolean;
}
```

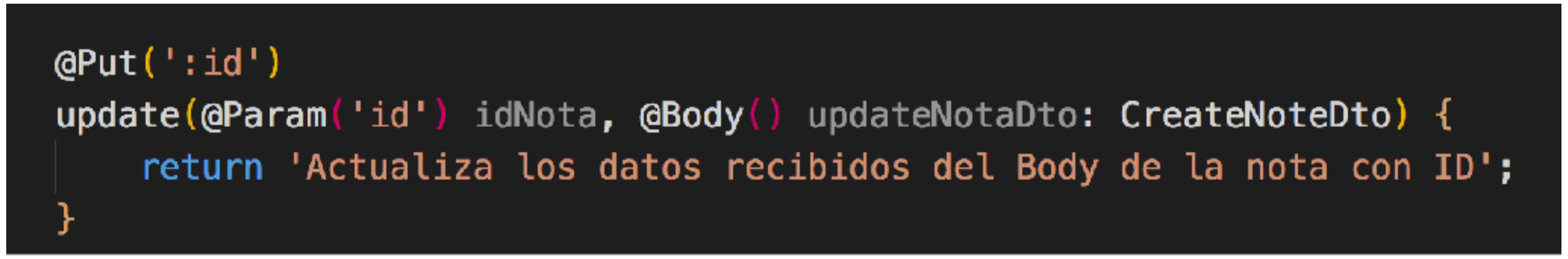
miguelcampos

2 CONTROLLERS

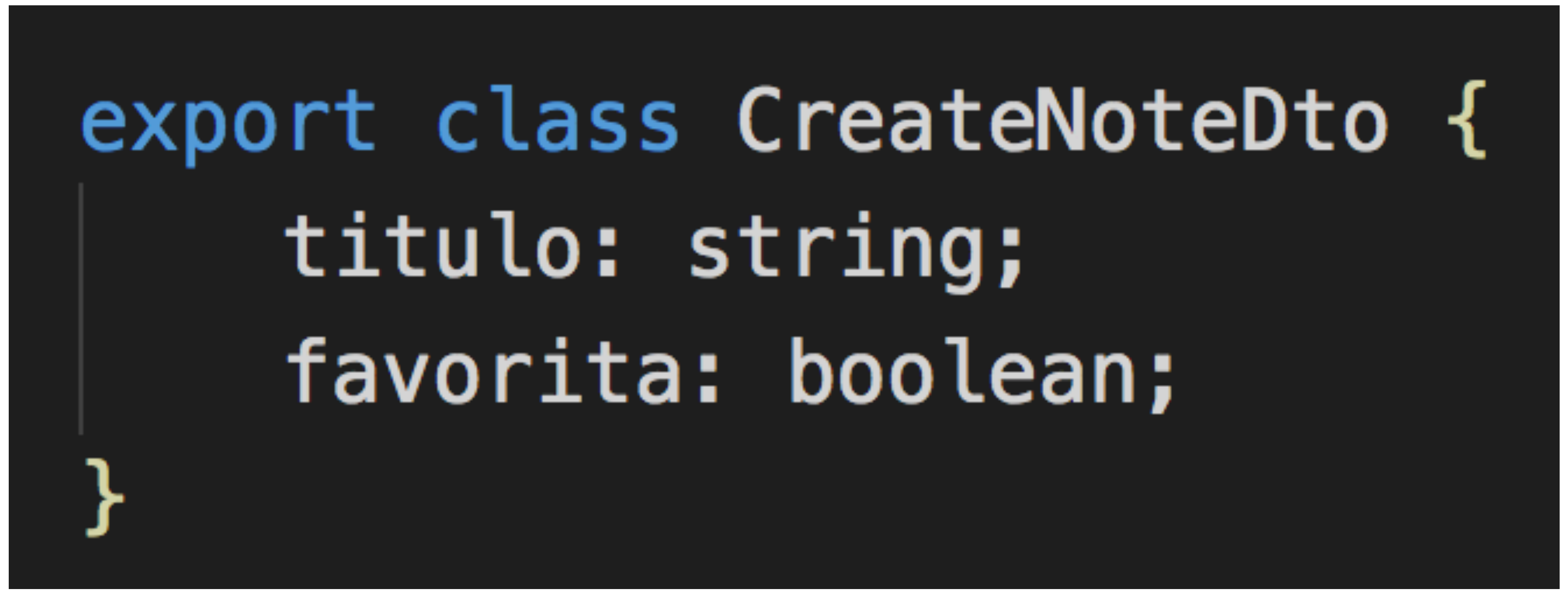
nest

PUT http://www.myapp.com/notes/1





```
@Put(':id')
update(@Param('id') idNota, @Body() updateNotaDto: CreateNoteDto) {
    return 'Actualiza los datos recibidos del Body de la nota con ID';
}
```

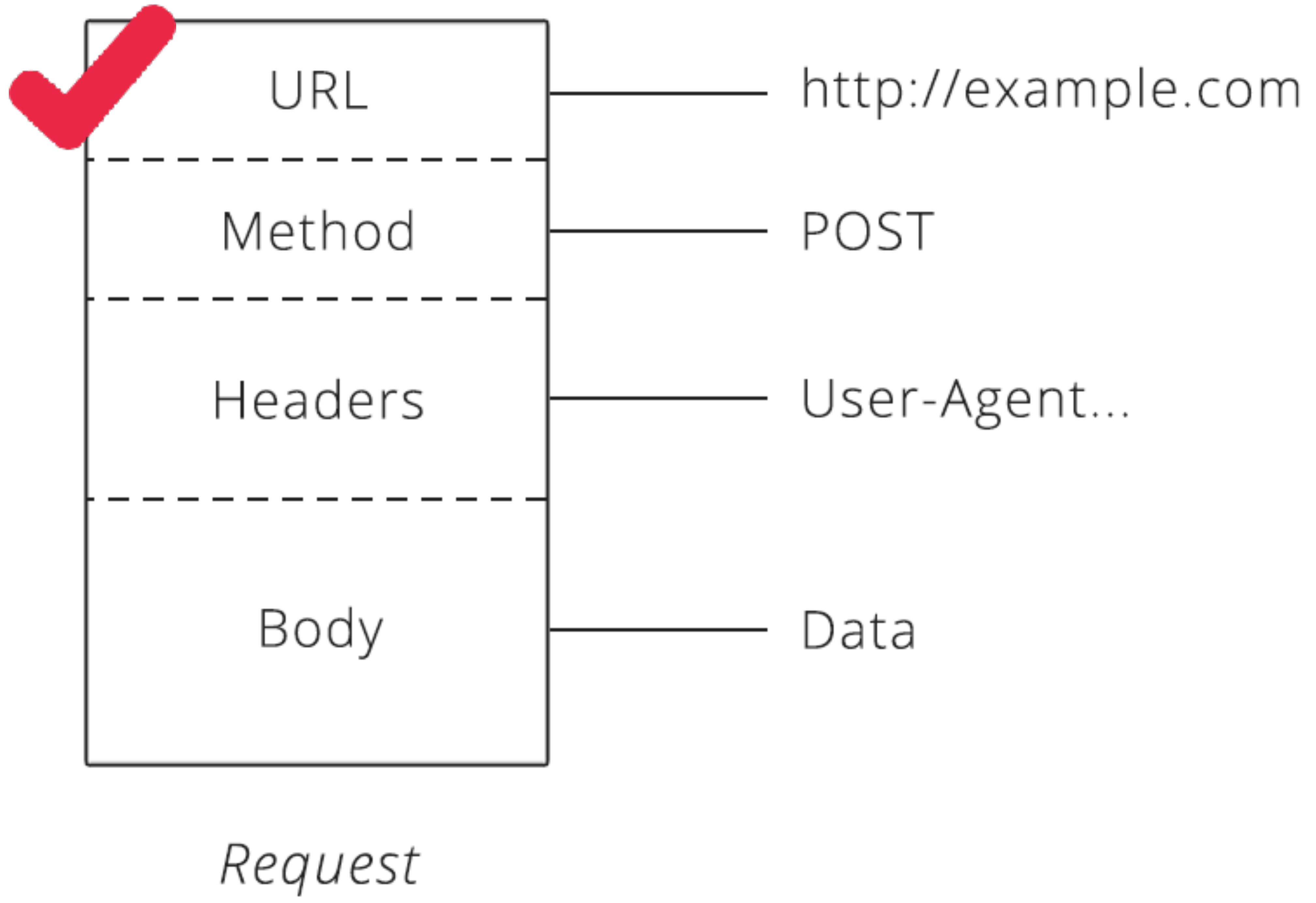


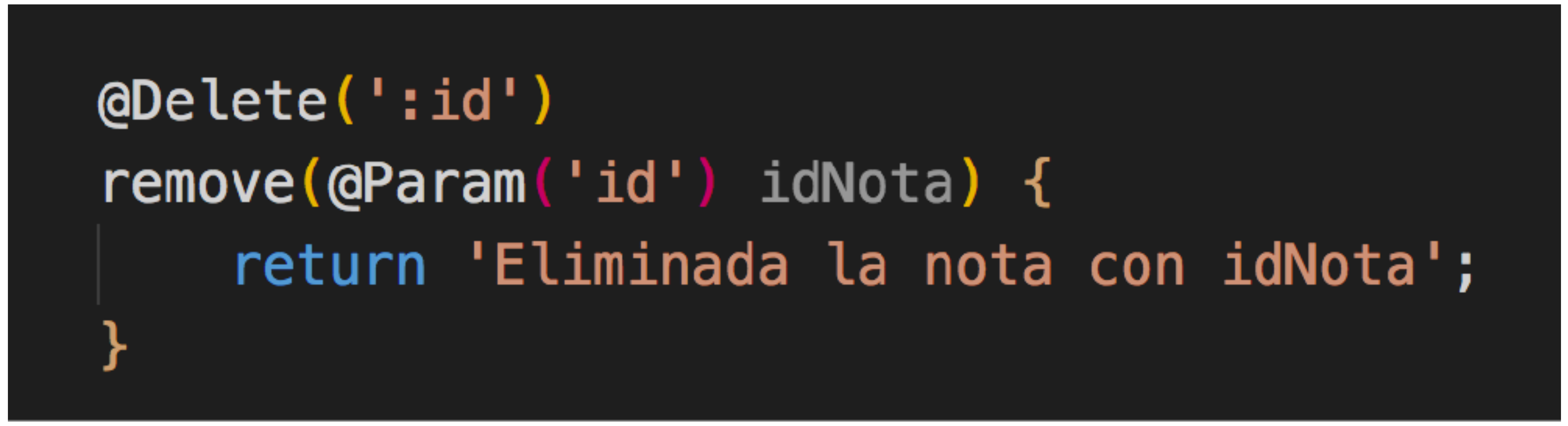
```
export class CreateNoteDto {
    titulo: string;
    favorita: boolean;
}
```

miguelcampos

2 CONTROLLERS

DEL http://www.myapp.com/notes/**1**

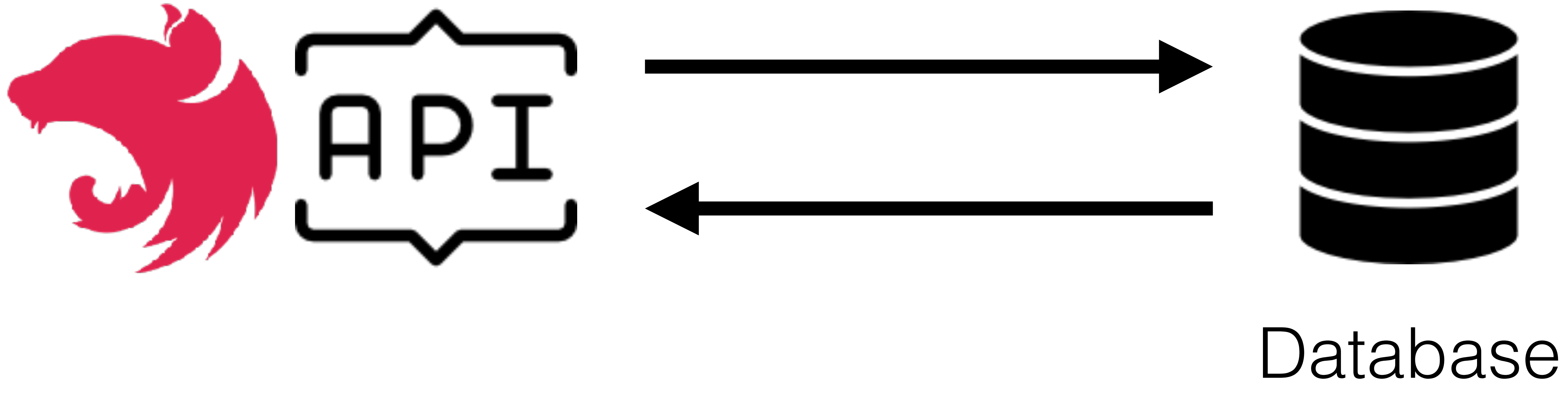




```
@Delete(':id')
remove(@Param('id') idNota) {
    return 'Eliminada la nota con idNota';
}
```

miguelcampos

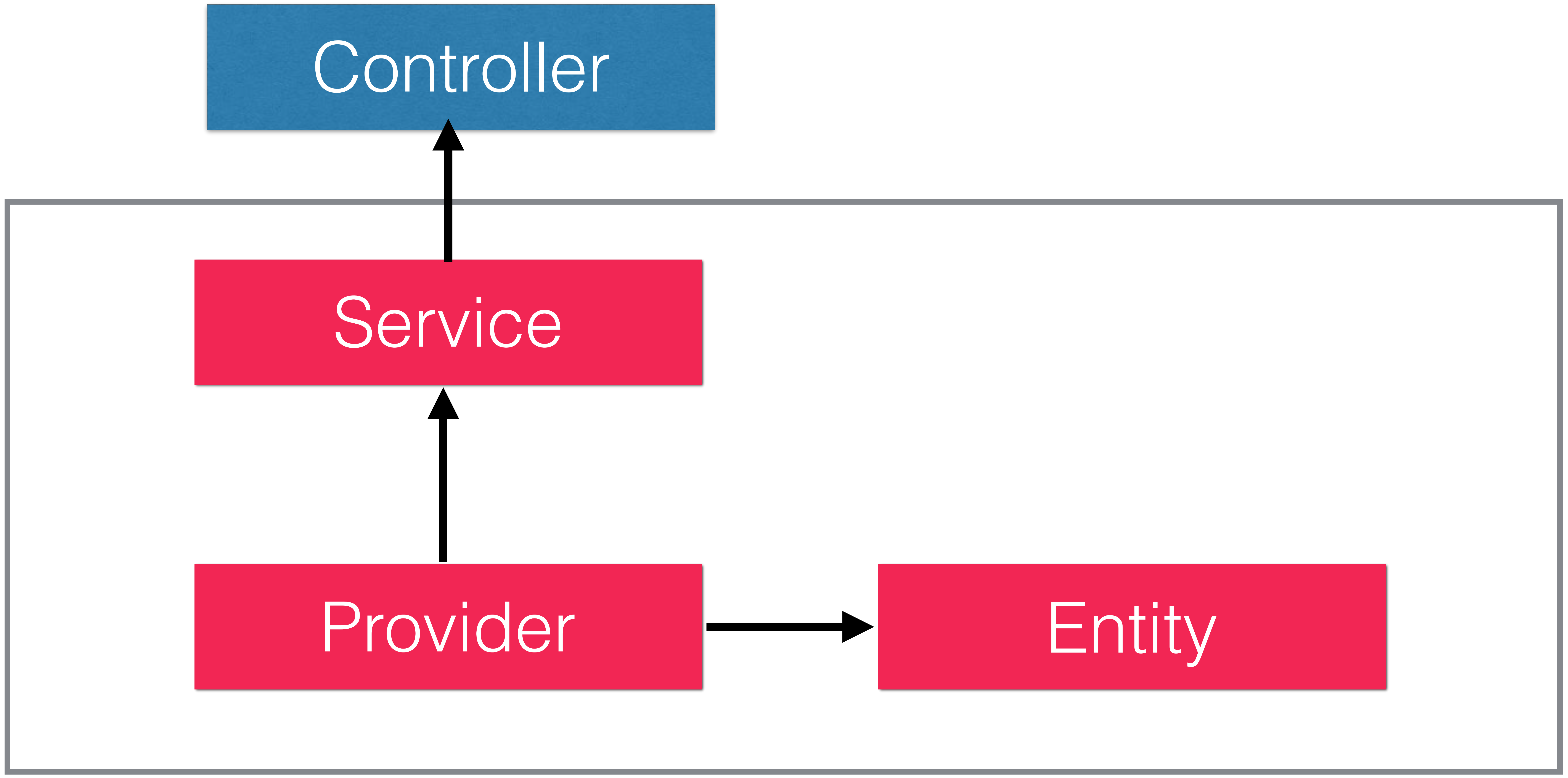
3 PROVIDERS



TYPEORM

TypeORM is an ORM that can run in NodeJS, Browser, Cordova, PhoneGap, Ionic, React Native, NativeScript, Expo, and Electron platforms and can be used with TypeScript and JavaScript (ES5, ES6, ES7, ES8).

3 PROVIDERS

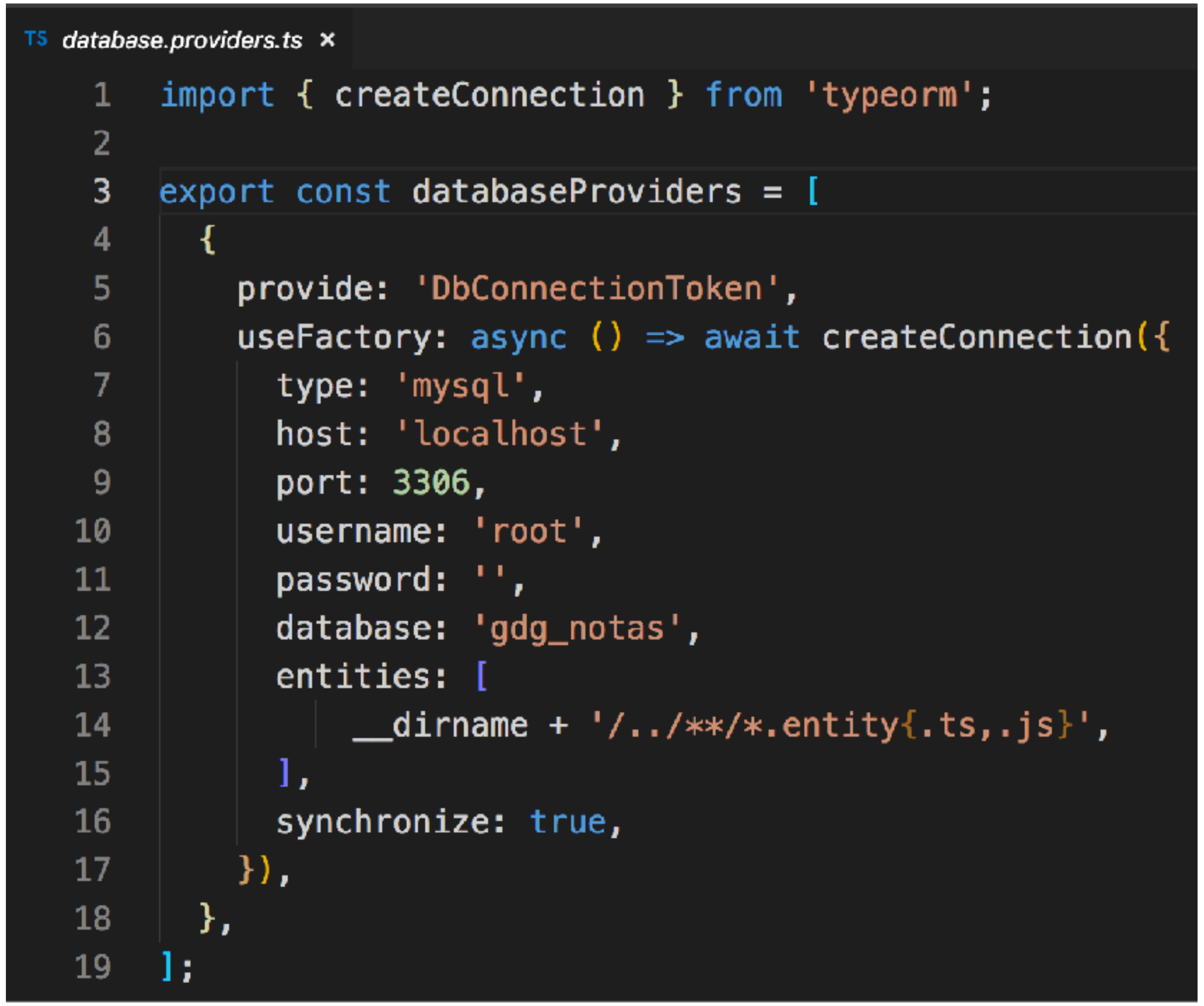


3 PROVIDERS

Instalar TypeORM

```
$ npm install --save typeorm mysql
```

Configuración conexión a la base de datos con **TypeORM:**



database.providers.ts

```
import { createConnection } from 'typeorm';

export const databaseProviders = [
  {
    provide: 'DbConnectionToken',
    useFactory: async () => await createConnection({
      type: 'mysql',
      host: 'localhost',
      port: 3306,
      username: 'root',
      password: '',
      database: 'gdg_notas',
      entities: [
        __dirname + '/../**/*.entity{.ts,.js}',
      ],
      synchronize: true,
    }),
  },
];
```

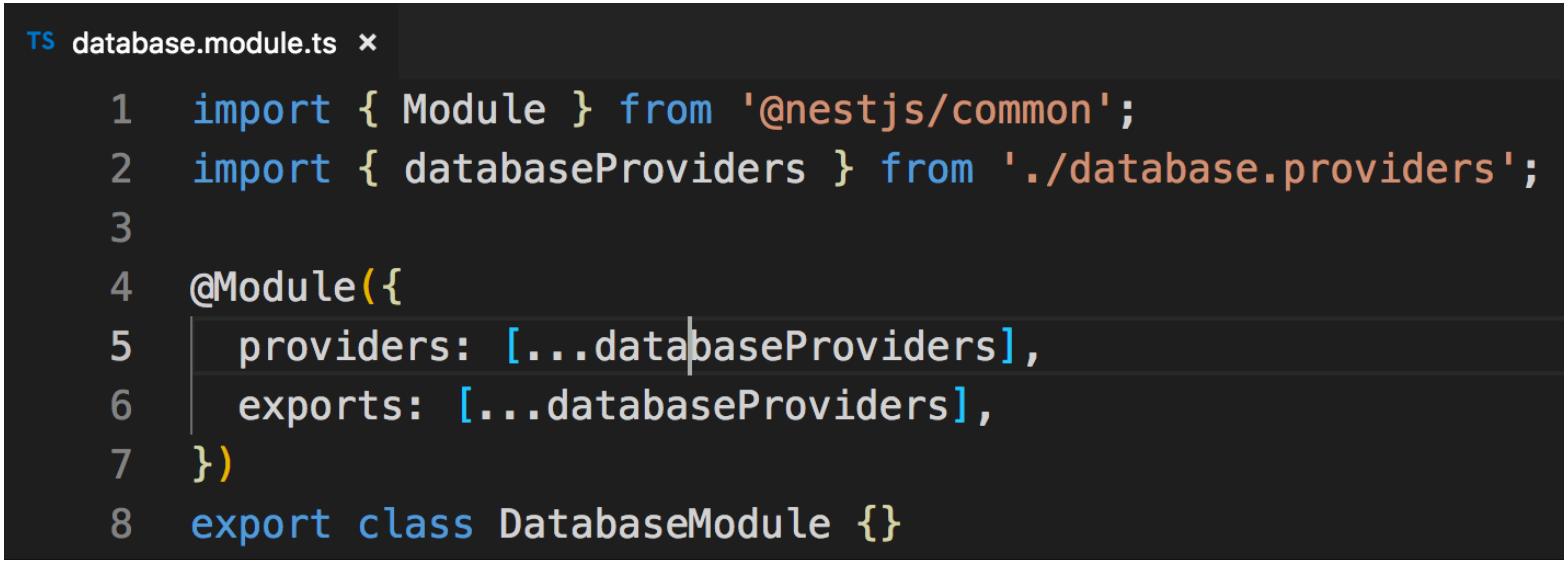
nest

miguelcampos

3 PROVIDERS

Creación Módulo

```
$ nest g module database
```



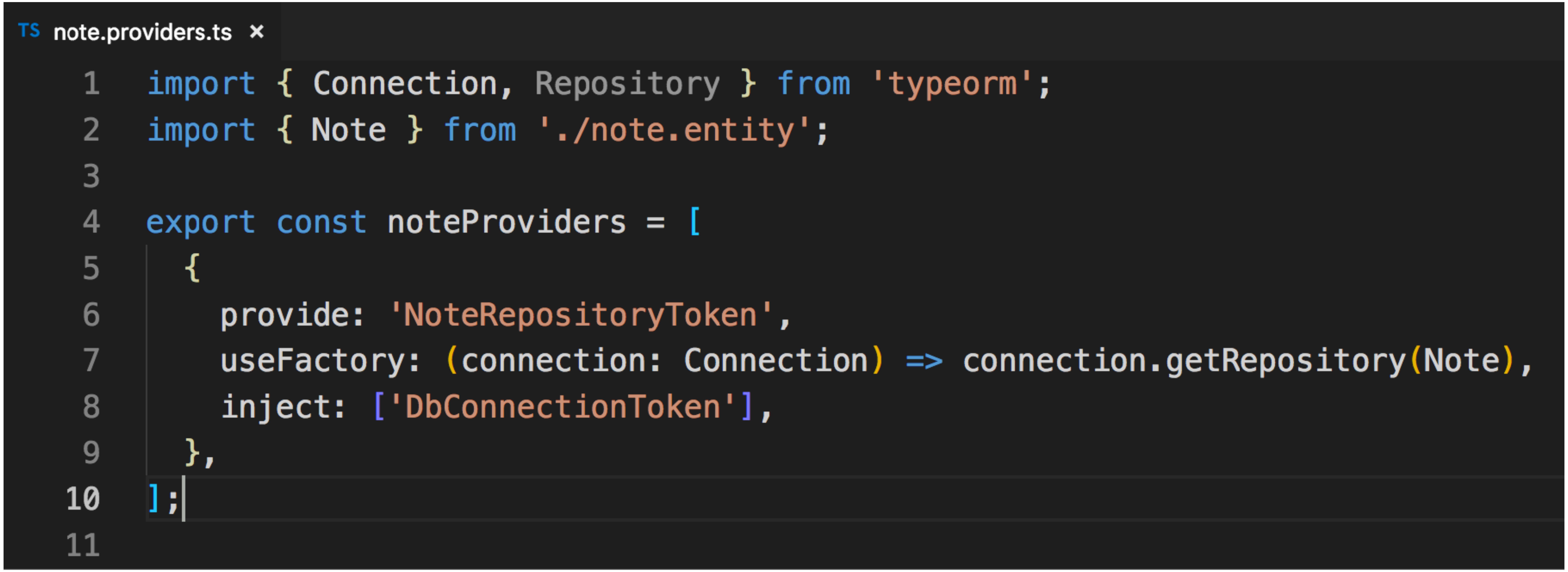
database.module.ts

```typescript
import { Module } from '@nestjs/common';
import { databaseProviders } from './database.providers';

@Module({
  providers: [...databaseProviders],
  exports: [...databaseProviders],
})
export class DatabaseModule {}
```

3

PROVIDERS

nest


note.providers.ts

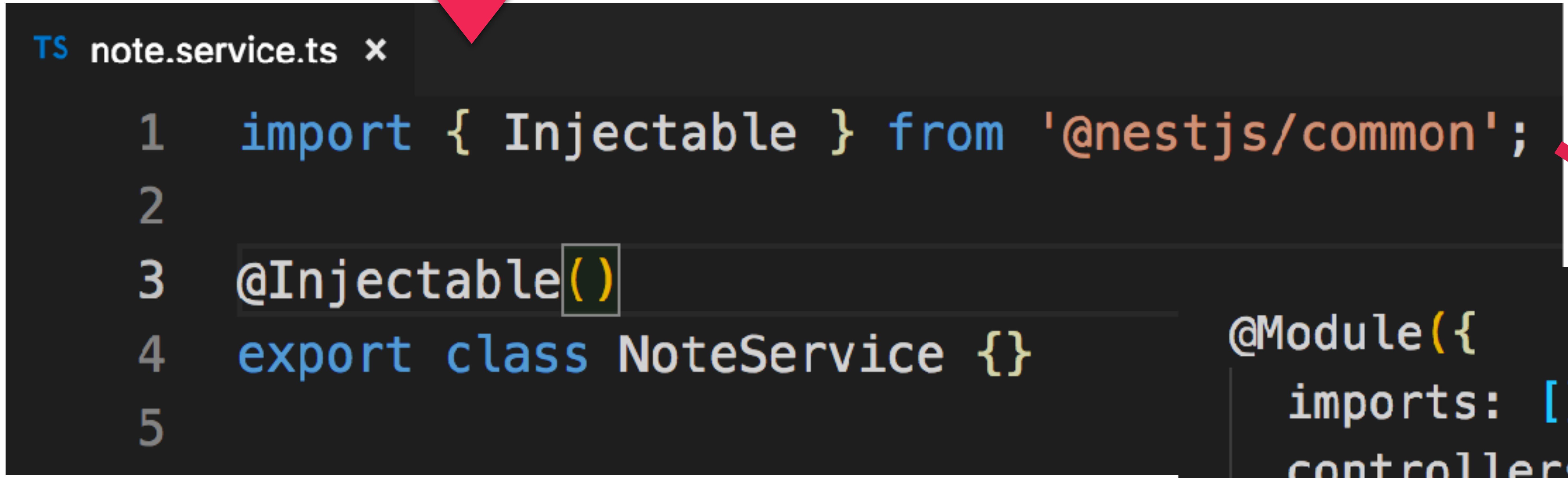
```typescript
import { Connection, Repository } from 'typeorm';
import { Note } from './note.entity';

export const noteProviders = [
  {
    provide: 'NoteRepositoryToken',
    useFactory: (connection: Connection) => connection.getRepository(Note),
    inject: ['DbConnectionToken'],
  },
];

```

miguelcampos

3 PROVIDERS

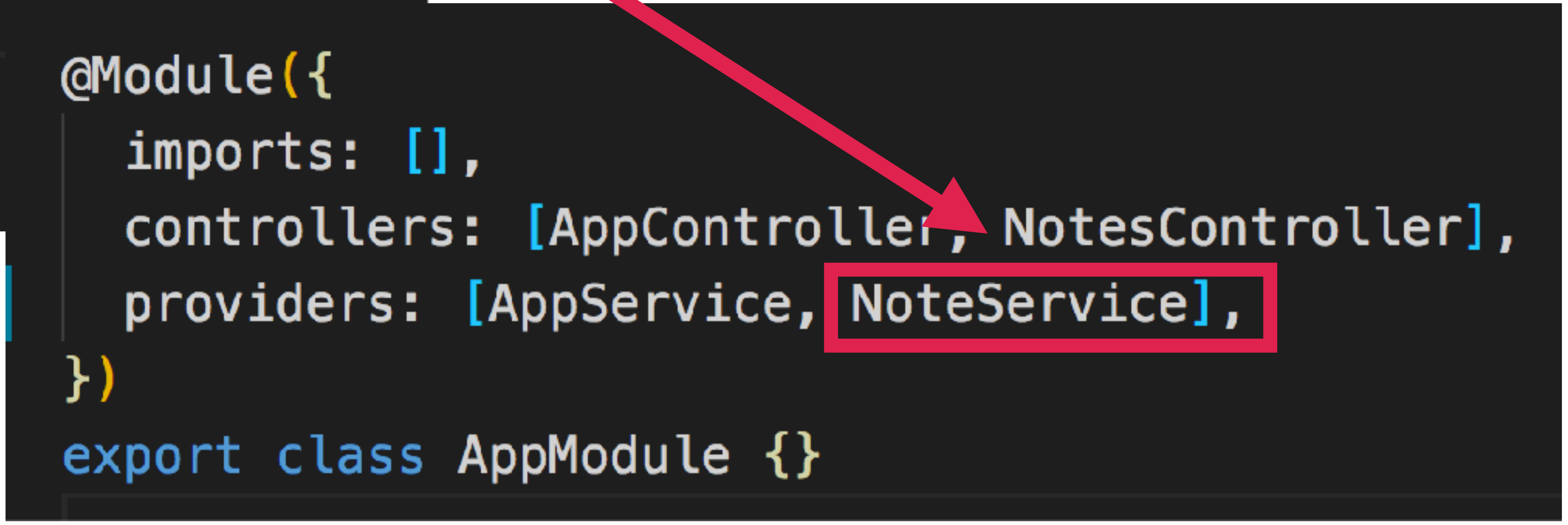
Creamos un Servicio para la gestión de la entidad **Note**:

```
$ nest g s notes/note
```



note.service.ts

```
import { Injectable } from '@nestjs/common';

@Injectable()
export class NoteService {}
```

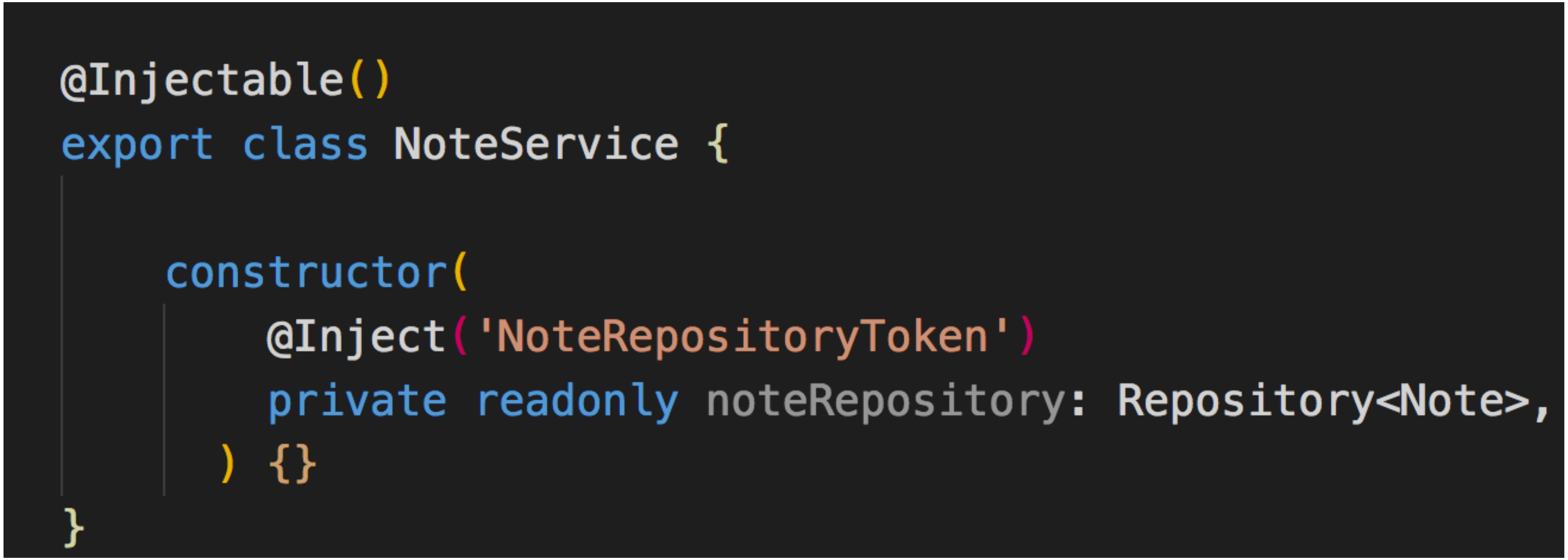


```
@Module({
  imports: [],
  controllers: [AppController, NotesController],
  providers: [AppService, NoteService],
})
export class AppModule {}
```

3 PROVIDERS

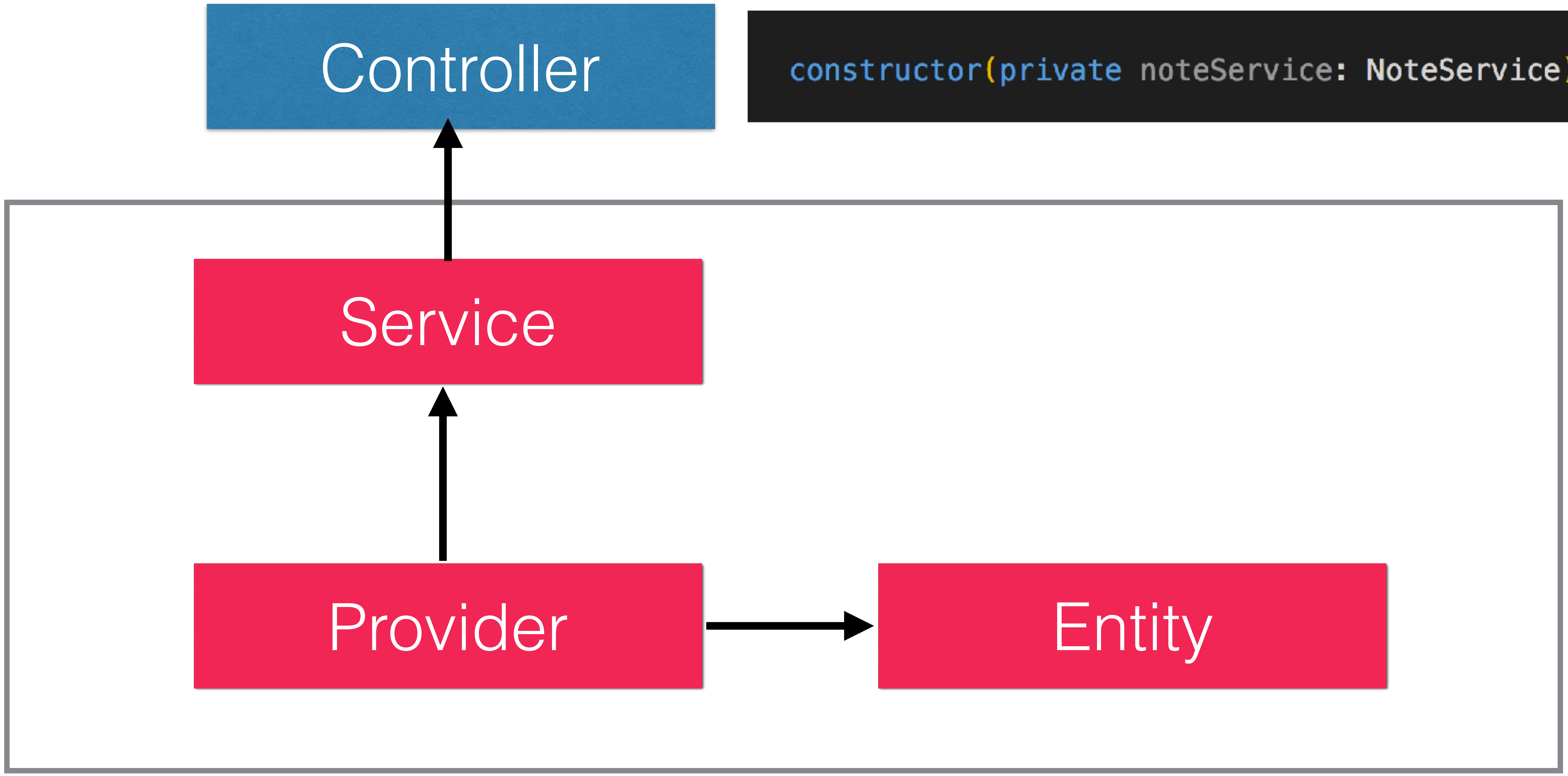
nest

Aplicamos la inyección de dependencias en el fichero ***note.service.ts***:



```
@Injectable()
export class NoteService {

    constructor(
        @Inject('NoteRepositoryToken')
        private readonly noteRepository: Repository<Note>,
    ) {}
}
```

3 PROVIDERS



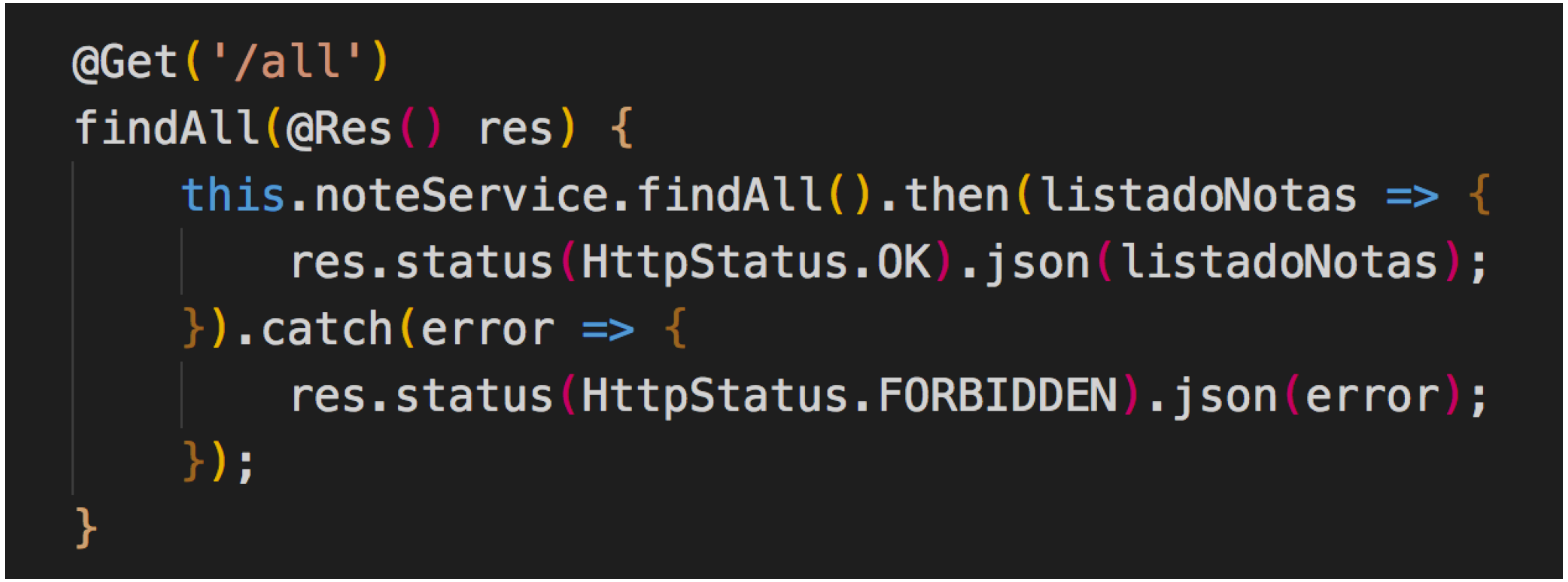
```
constructor(private noteService: NoteService) {}
```

3 PROVIDERS

nest

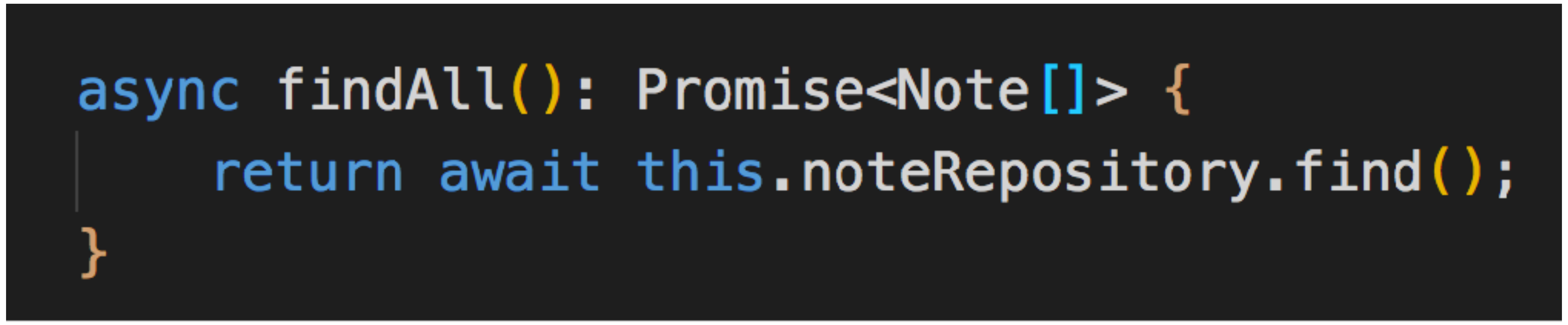
Listar todas las notas, petición GET:

note.controller.ts:



```
@Get('/all')
findAll(@Res() res) {
    this.noteService.findAll().then(listadoNotas => {
        res.status(HttpStatus.OK).json(listadoNotas);
    }).catch(error => {
        res.status(HttpStatus.FORBIDDEN).json(error);
    });
}
```

note.service.ts:



```
async findAll(): Promise<Note[]> {
    return await this.noteRepository.find();
}
```

miguelcampos

3 PROVIDERS

nest

Listar una nota, petición GET, parámetro ID:

note.controller.ts:

```
@Get(':id')
findOne(@Param('id') idNota, @Res() res) {
    this.noteService.findOne(idNota).then(nota => {
        res.status(HttpStatus.OK).json(nota);
    }).catch(error => {
        res.status(HttpStatus.FORBIDDEN).json(error);
    });
}
```

note.service.ts:

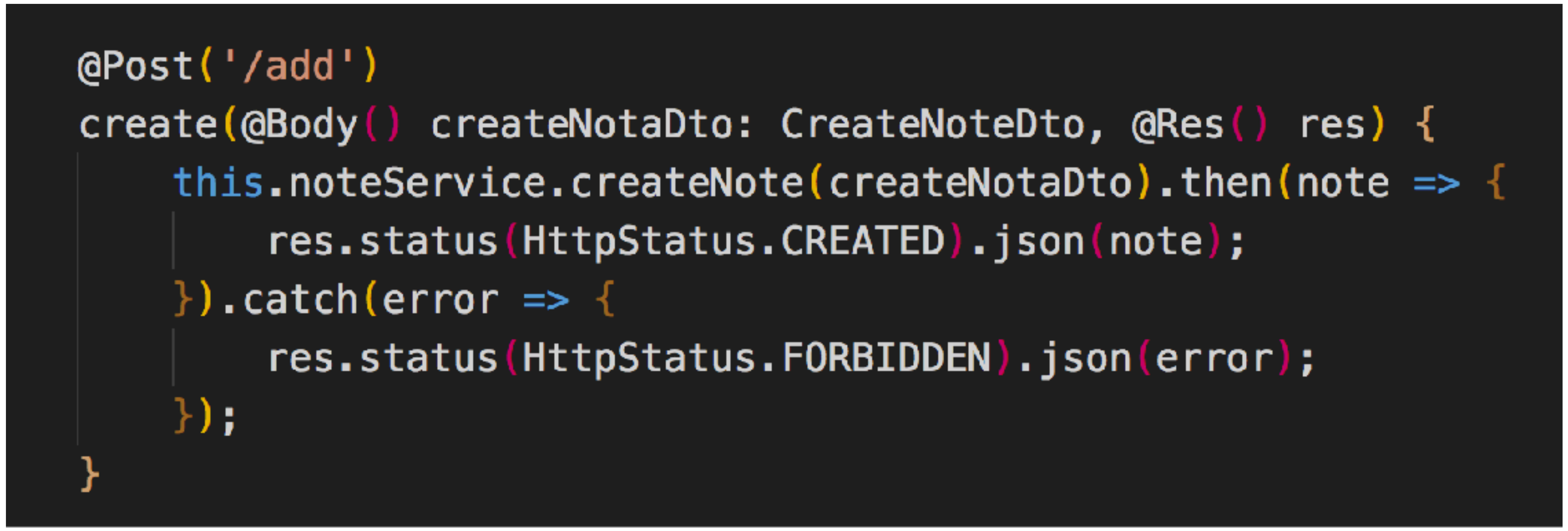
```
async findOne(idNota: number): Promise<Note> {
    return await this.noteRepository.findOne(idNota);
}
```

3 PROVIDERS

nest

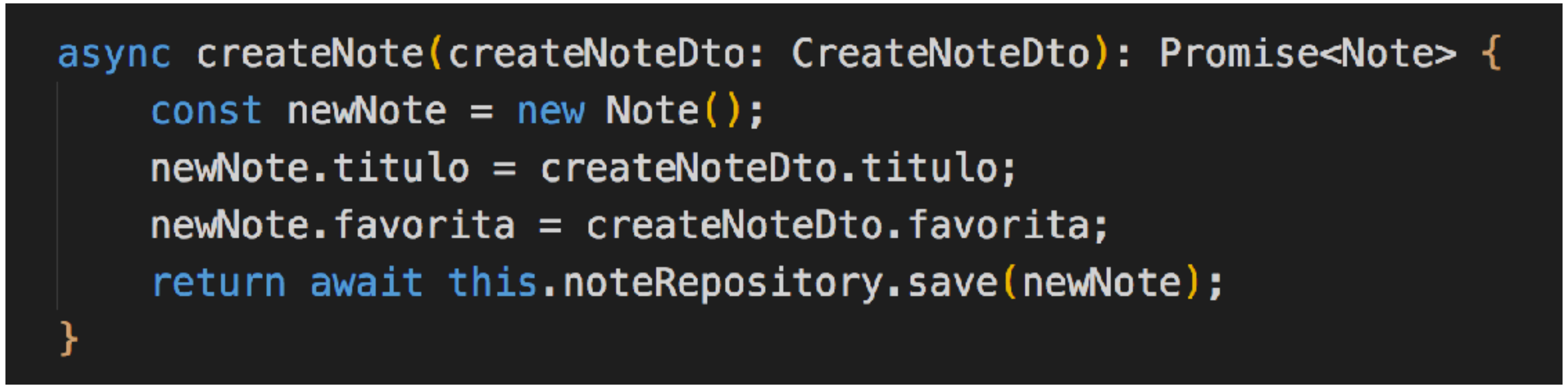
Añadir una nueva nota, POST, datos en Body:

note.controller.ts:



```
@Post('/add')
create(@Body() createNotaDto: CreateNoteDto, @Res() res) {
    this.noteService.createNote(createNotaDto).then(note => {
        res.status(HttpStatus.CREATED).json(note);
    }).catch(error => {
        res.status(HttpStatus.FORBIDDEN).json(error);
    });
}
```

note.service.ts:



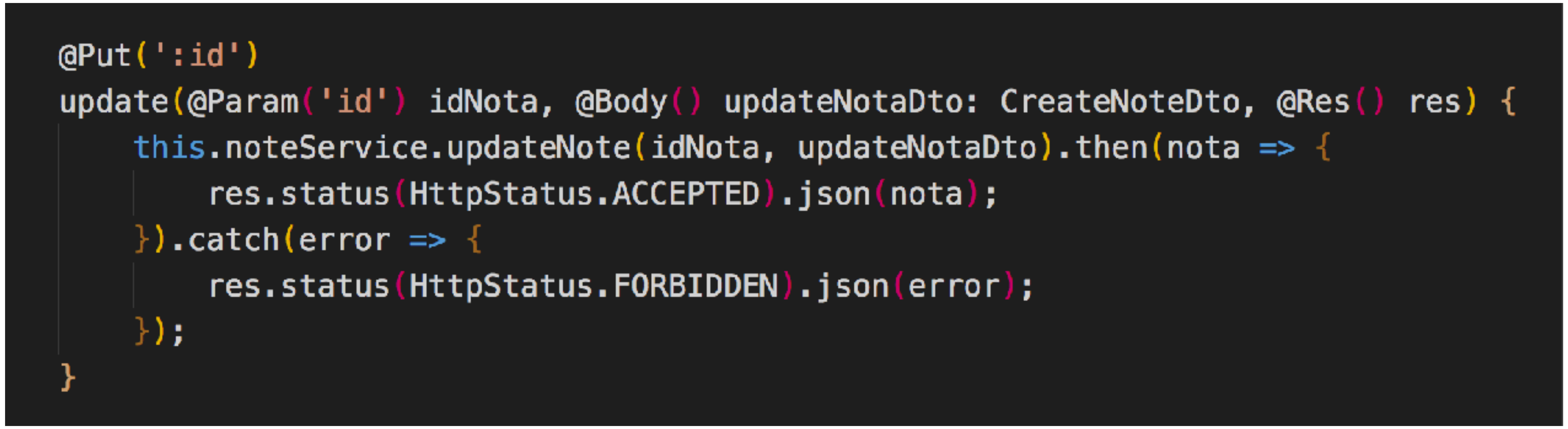
```
async createNote(createNoteDto: CreateNoteDto): Promise<Note> {
    const newNote = new Note();
    newNote.titulo = createNoteDto.titulo;
    newNote.favorita = createNoteDto.favorita;
    return await this.noteRepository.save(newNote);
}
```

3 PROVIDERS

nest

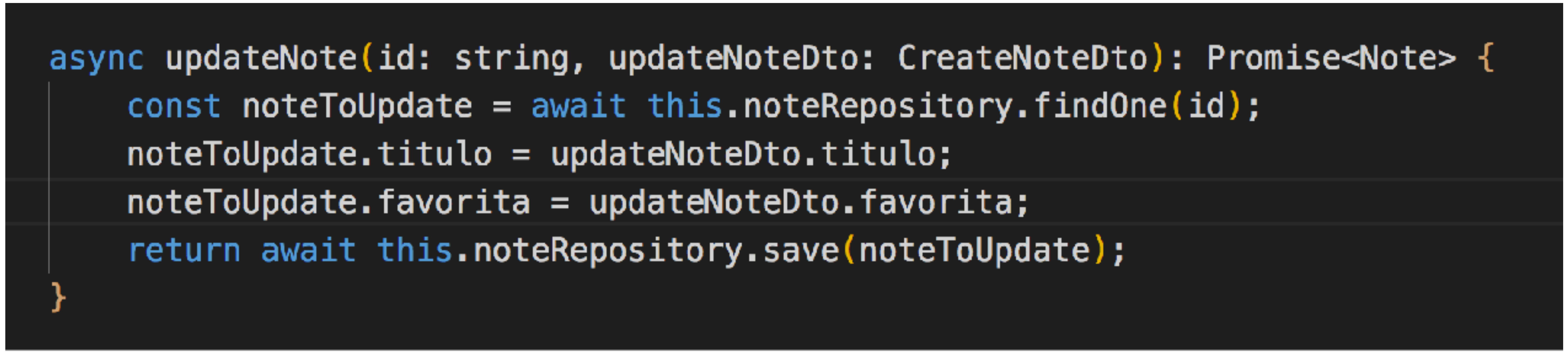
Actualizar los datos de una nota, PUT, param URL, datos Body:

note.controller.ts:



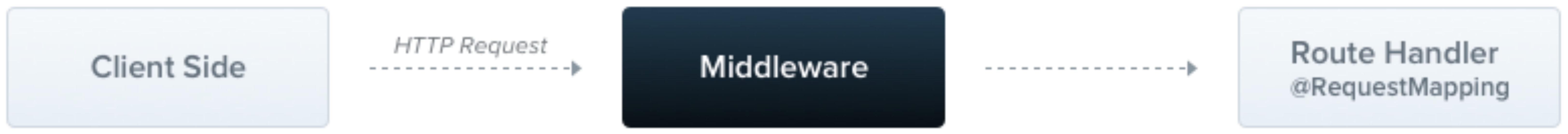
```
@Put(':id')
update(@Param('id') idNota, @Body() updateNotaDto: CreateNoteDto, @Res() res) {
    this.noteService.updateNote(idNota, updateNotaDto).then(nota => {
        res.status(HttpStatus.ACCEPTED).json(nota);
    }).catch(error => {
        res.status(HttpStatus.FORBIDDEN).json(error);
    });
}
```

note.service.ts:

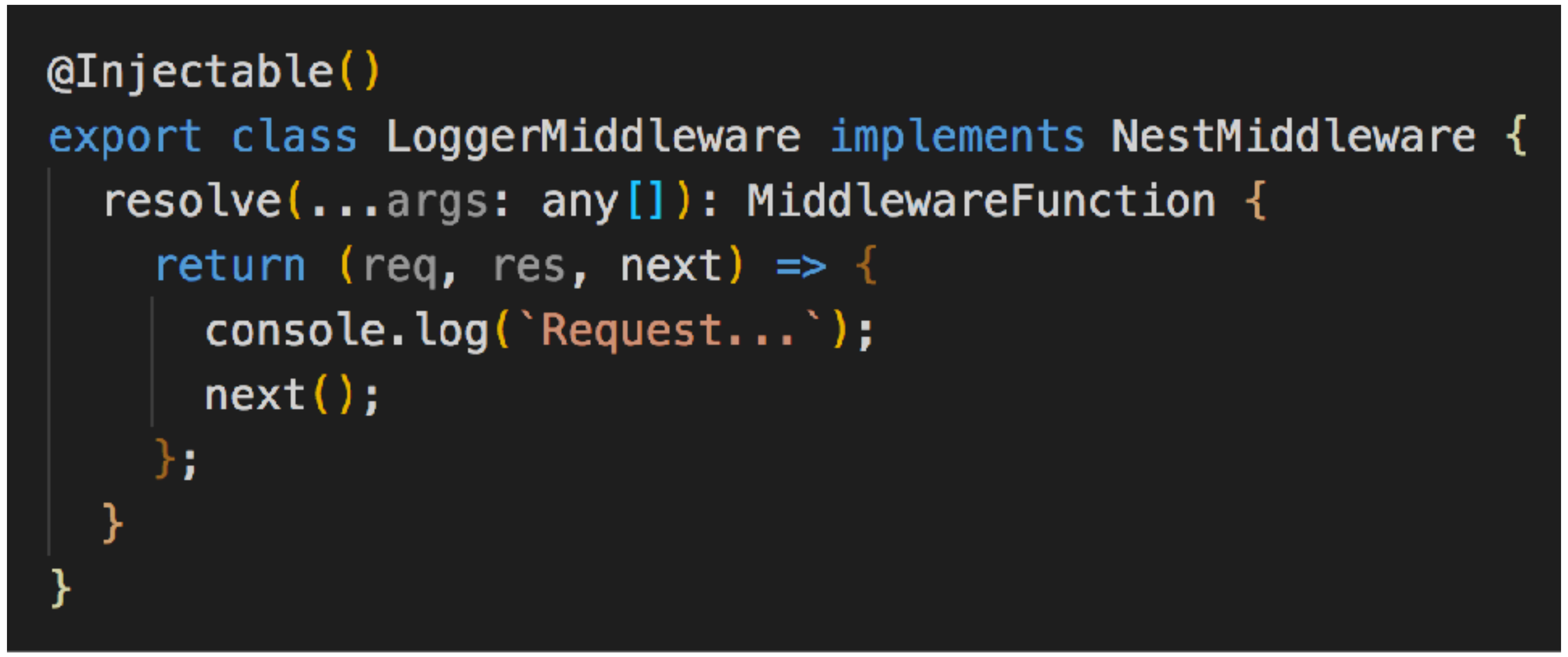


```
async updateNote(id: string, updateNoteDto: CreateNoteDto): Promise<Note> {
    const noteToUpdate = await this.noteRepository.findOne(id);
    noteToUpdate.titulo = updateNoteDto.titulo;
    noteToUpdate.favorita = updateNoteDto.favorita;
    return await this.noteRepository.save(noteToUpdate);
}
```

5 EXTRAS - MIDDLEWARES

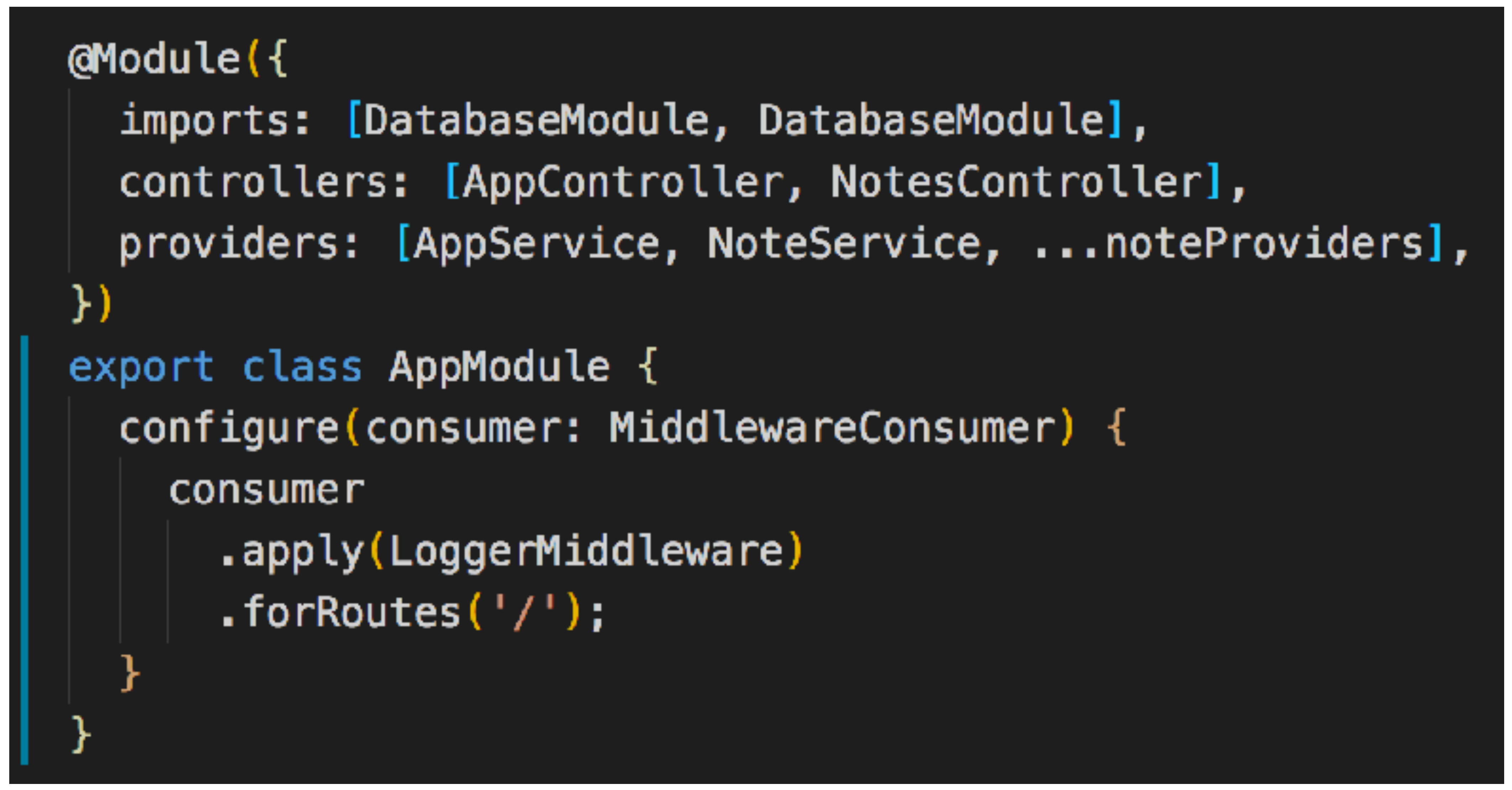


Definición del Middleware ***logger.middleware.ts***



```
@Injectable()
export class LoggerMiddleware implements NestMiddleware {
  resolve(...args: any[]): MiddlewareFunction {
    return (req, res, next) => {
      console.log(`Request...`);
      next();
    };
  }
}
```

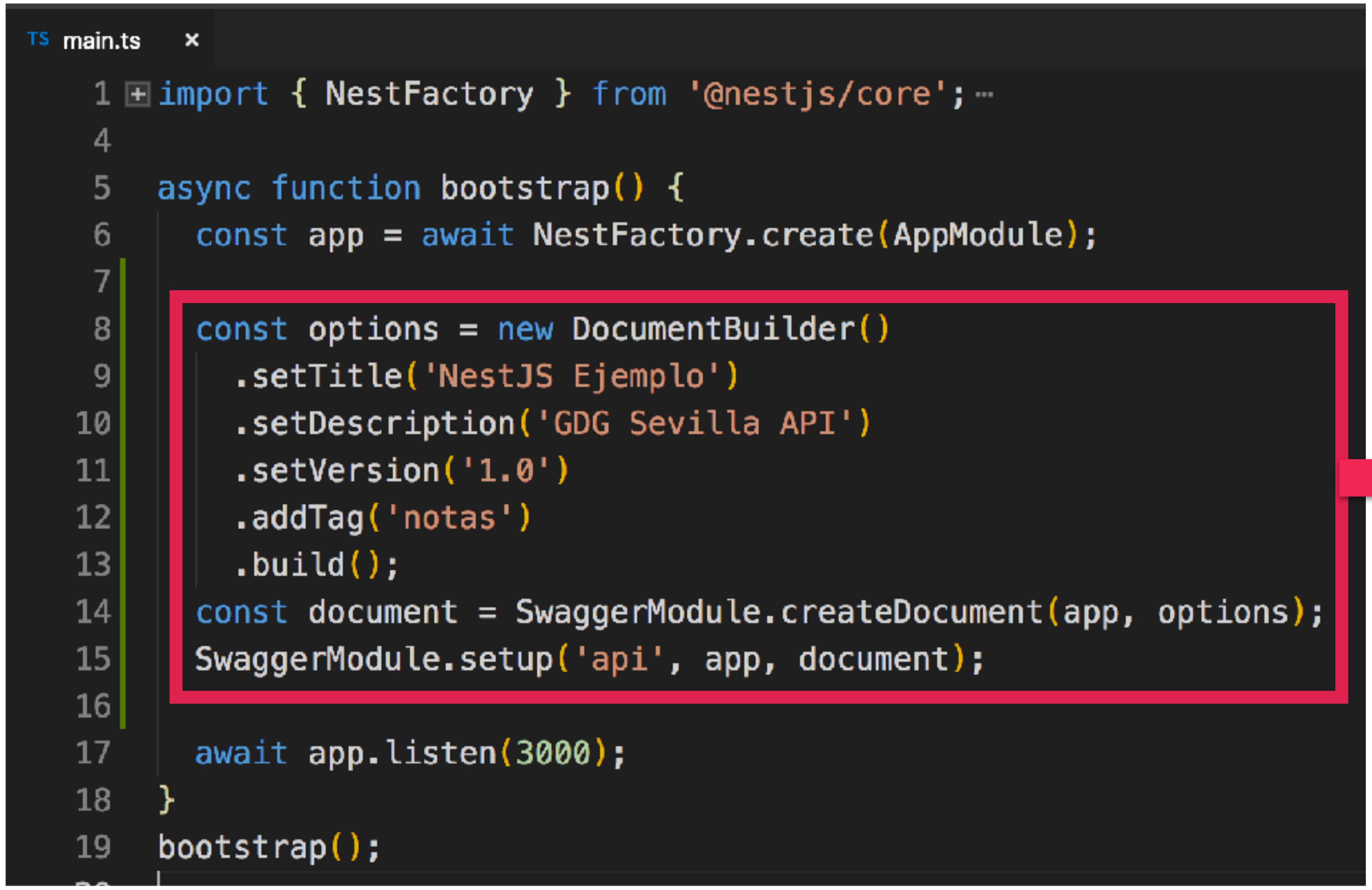
Configuración del Middleware en el AppModule para todas las rutas "/"



```
@Module({
  imports: [DatabaseModule, DatabaseModule],
  controllers: [AppController, NotesController],
  providers: [AppService, NoteService, ...noteProviders],
})
export class AppModule {
  configure(consumer: MiddlewareConsumer) {
    consumer
      .apply(LoggerMiddleware)
      .forRoutes('/');
  }
}
```

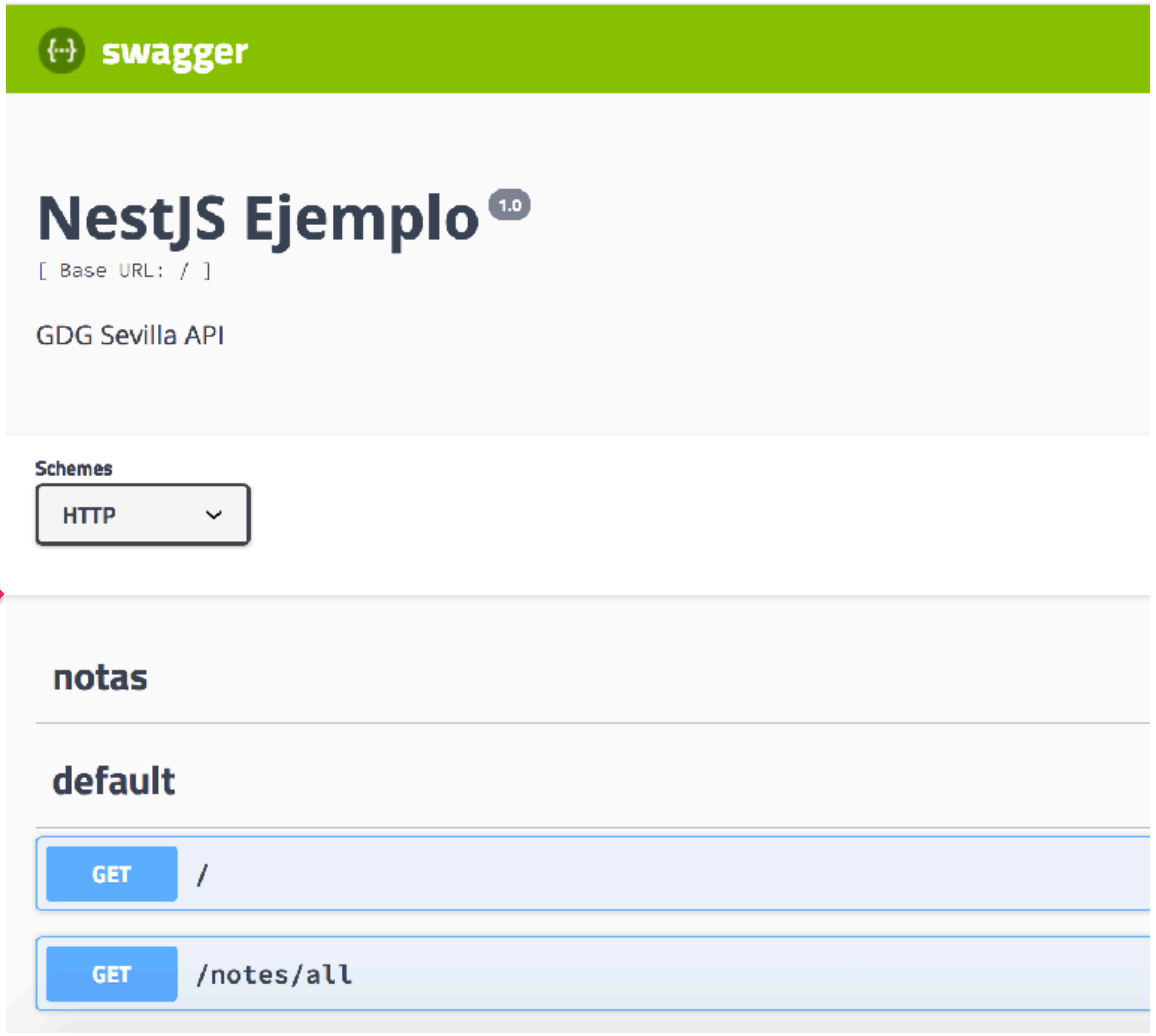
4 EXTRAS - SWAGGER

```
$ npm install --save @nestjs/swagger
```



main.ts

```
import { NestFactory } from '@nestjs/core';…

async function bootstrap() {
  const app = await NestFactory.create(AppModule);

  const options = new DocumentBuilder()
    .setTitle('NestJS Ejemplo')
    .setDescription('GDG Sevilla API')
    .setVersion('1.0')
    .addTag('notas')
    .build();
  const document = SwaggerModule.createDocument(app, options);
  SwaggerModule.setup('api', app, document);

  await app.listen(3000);
}
bootstrap();
```



swagger

NestJS Ejemplo 1.0

[Base URL: /]

GDG Sevilla API

Schemes: HTTP

notas

default

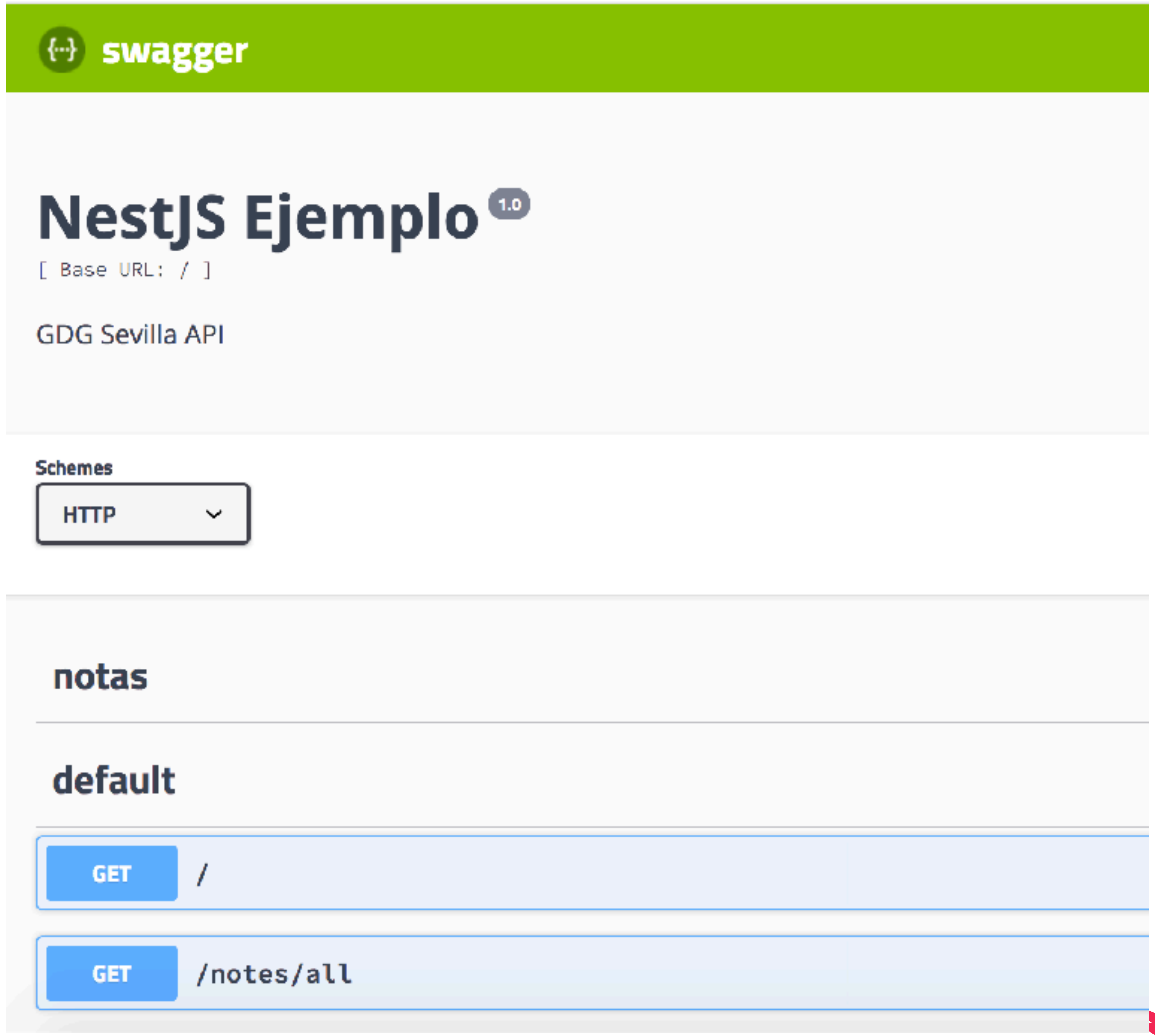
GET /

GET /notes/all

4 EXTRAS - SWAGGER

nest

http://localhost:3000/api


swagger

NestJS Ejemplo 1.0

[Base URL: /]

GDG Sevilla API

Schemes

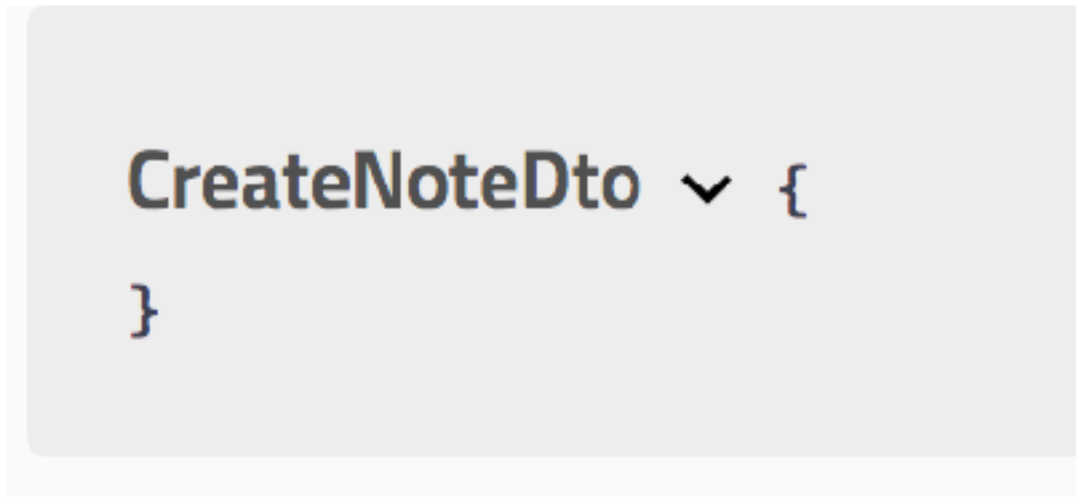
HTTP

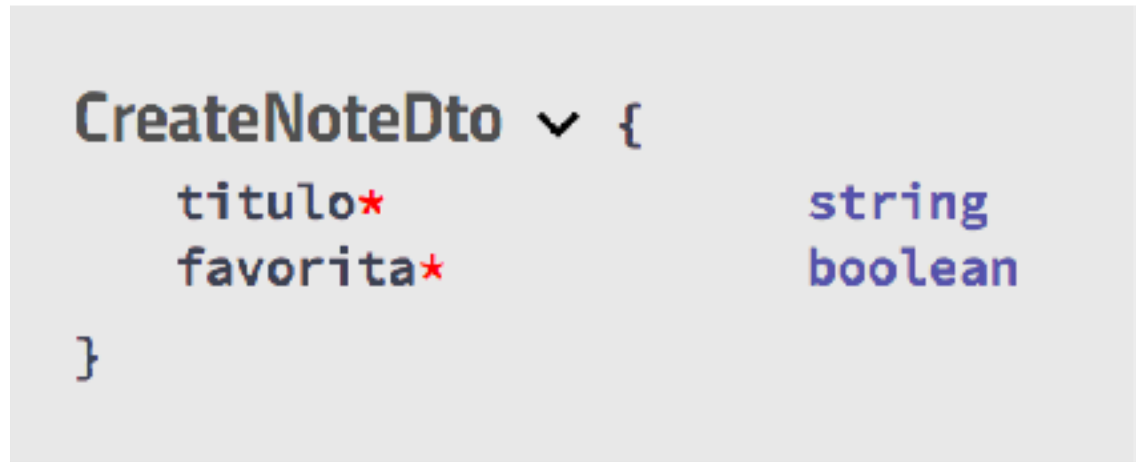
notas

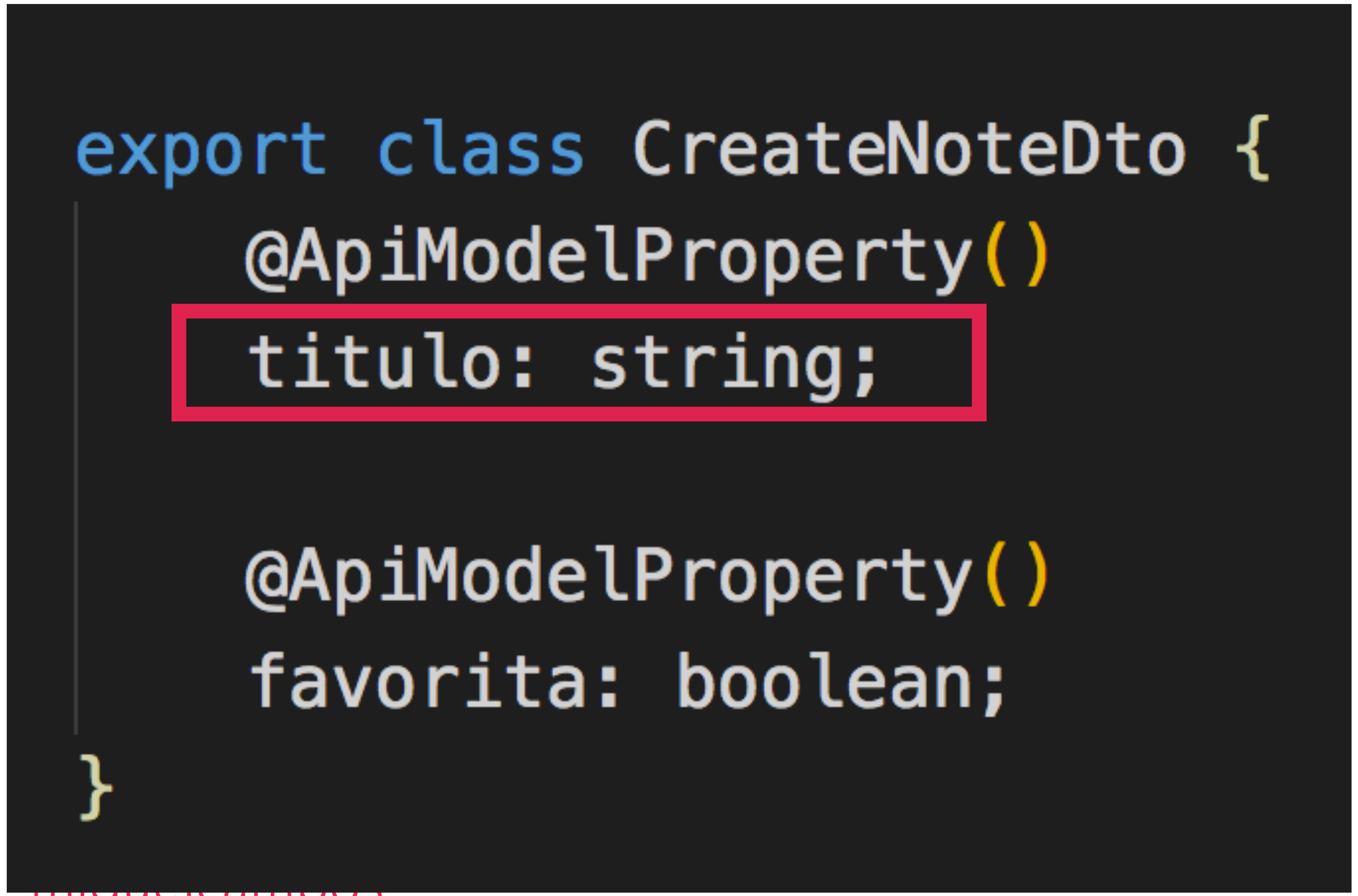
default

GET /

GET /notes/all


```
CreateNoteDto ∨ {
}
```


```
CreateNoteDto ∨ {
    titulo*          string
    favorita*        boolean
}
```

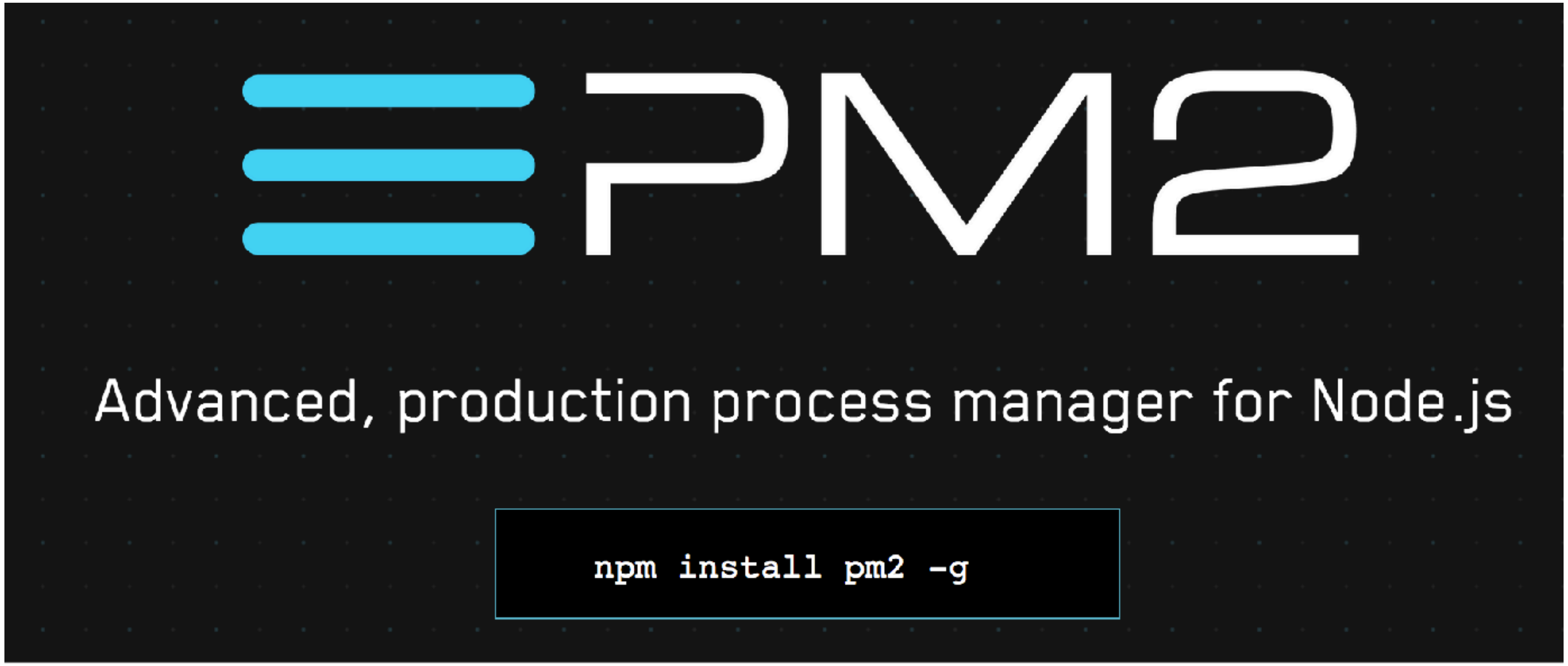

```
export class CreateNoteDto {
    @ApiModelProperty()
    titulo: string;

    @ApiModelProperty()
    favorita: boolean;
}
```

miguelcampos

5 RUN - DEPLOY

nest

```
$ npm run start
```



Advanced, production process manager for Node.js

```
npm install pm2 -g
```

¡GRACIAS!

"The only way to learn a new programming language is by writing programs in it."
Dennis Ritchie

miguelcampos camposmiguel