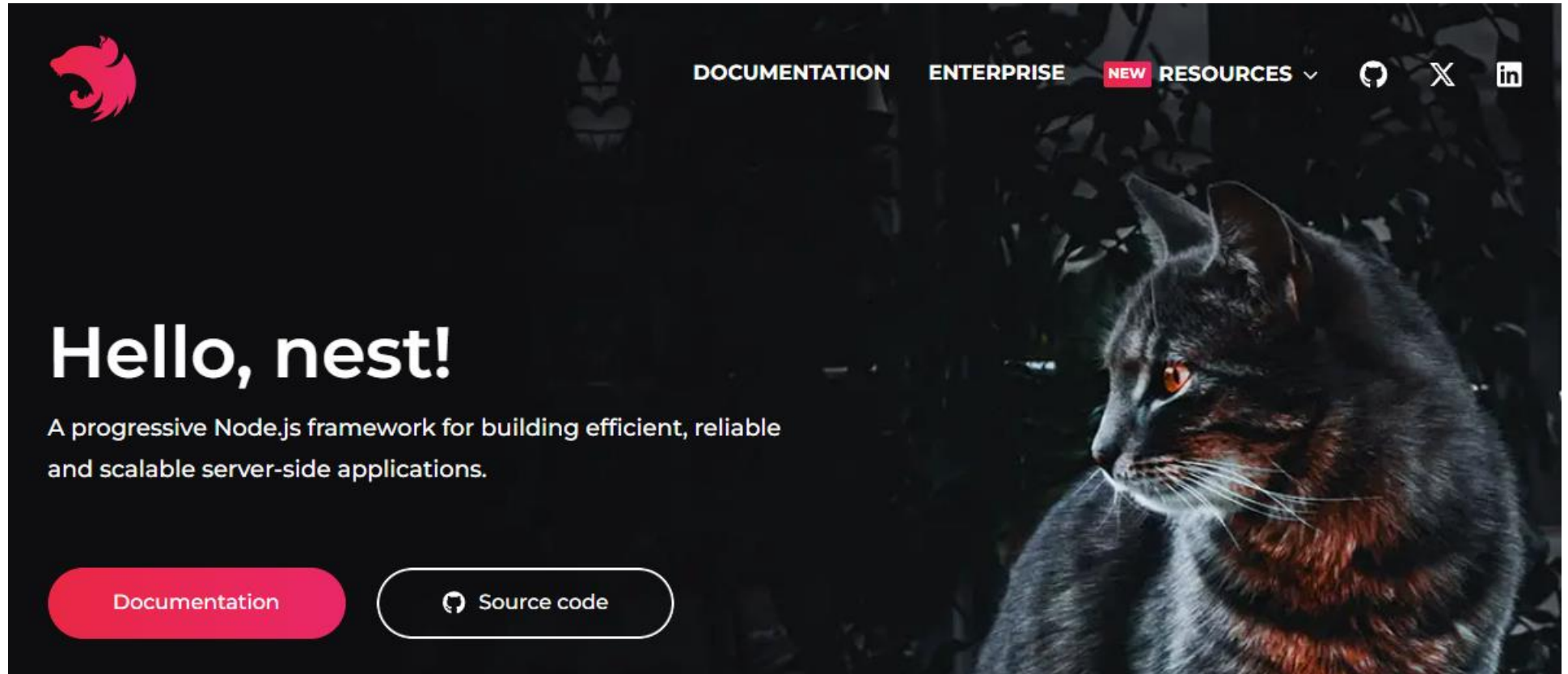


Nest.js



Prof. Dr. Robyson Aggio

www.aggioirati.com.br

Introdução



Nest JS



Estrutura e Arquitetura



Modules, Providers, Exports, Imports



Controllers / Services



DTO's, Pipes



Prisma ORM, Banco de Dados



Middlewares, Filters,
Interceptors, guards



Autenticação



Upload Arquivos



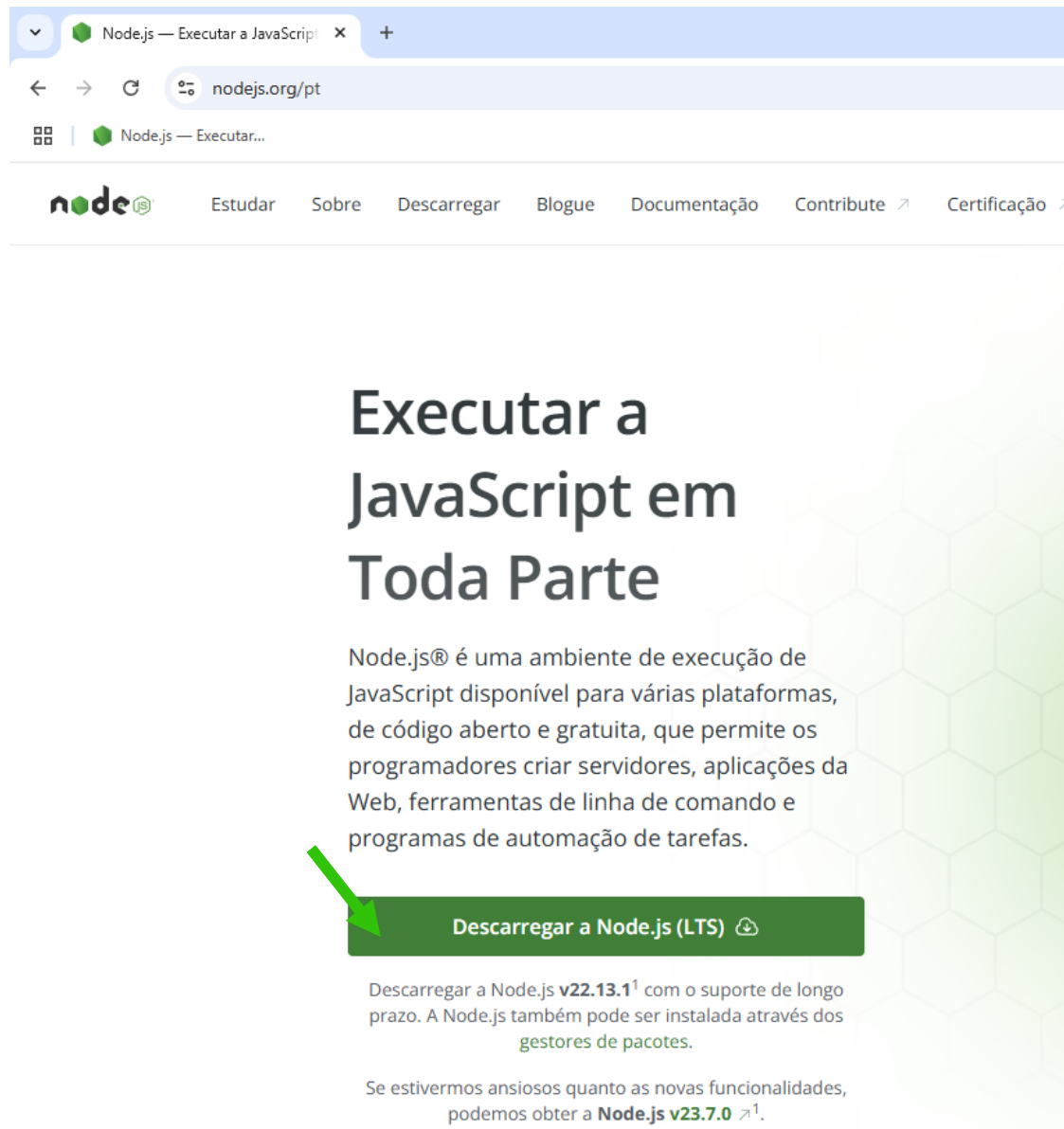
Melhorando Arquitetura



Testes Automatizados



Swagger



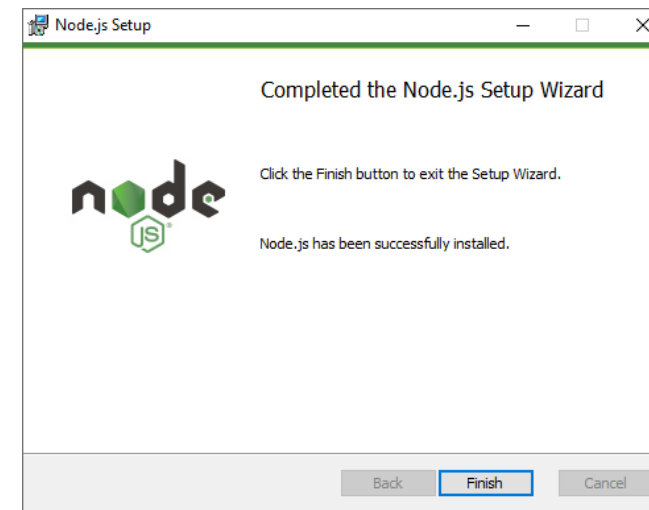
Executar a JavaScript em Toda Parte

Node.js® é uma ambiente de execução de JavaScript disponível para várias plataformas, de código aberto e gratuita, que permite os programadores criar servidores, aplicações da Web, ferramentas de linha de comando e programas de automação de tarefas.

[Descarregar a Node.js \(LTS\)](#)

Descarregar a Node.js **v22.13.1** com o suporte de longo prazo. A Node.js também pode ser instalada através dos gestores de pacotes.

Se estivermos ansiosos quanto as novas funcionalidades, podemos obter a **Node.js v23.7.0**.



```
C:\Windows\System32\cmd.exe
Microsoft Windows [versão 10.0.19045.5371]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Aggio\nestjs>node -v
v22.13.1

C:\Aggio\nestjs>npm -v
10.9.2

C:\Aggio\nestjs>
```

npm = Node Package Manager



 **Visual Studio Code** [Docs](#) [Updates](#) [Blog](#) [API](#) [Extensions](#) [FAQ](#) [GitHub Copilot](#)



[Download](#)

 Get [GitHub Copilot Free](#) in VS Code!

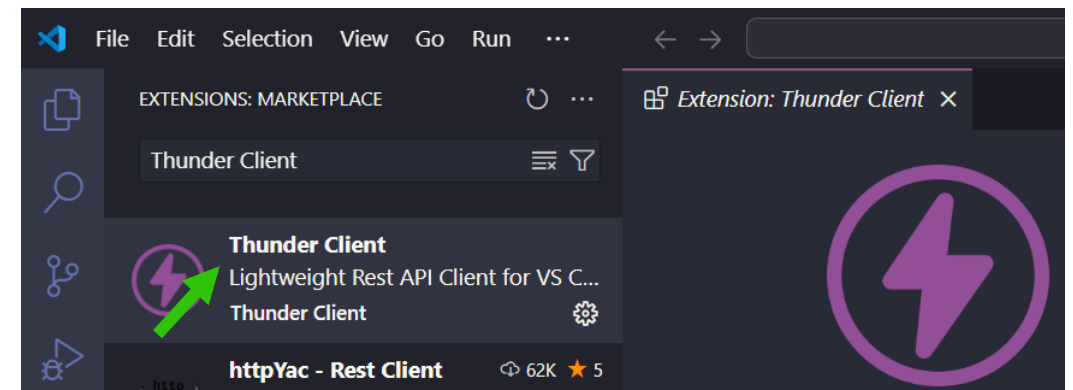
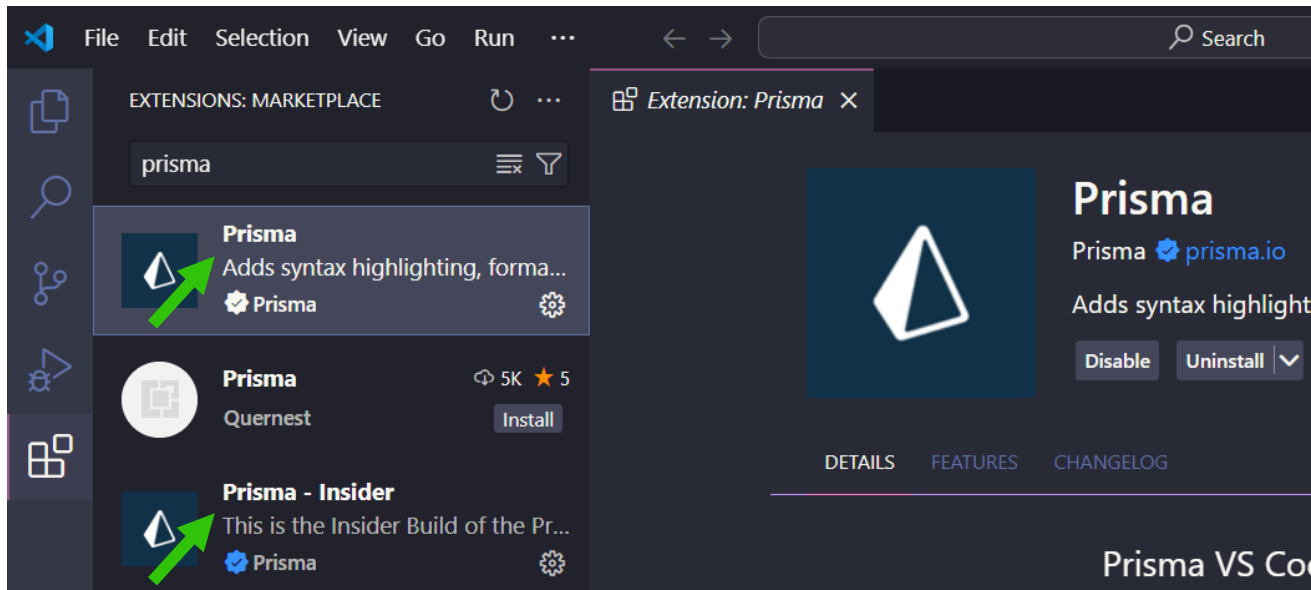
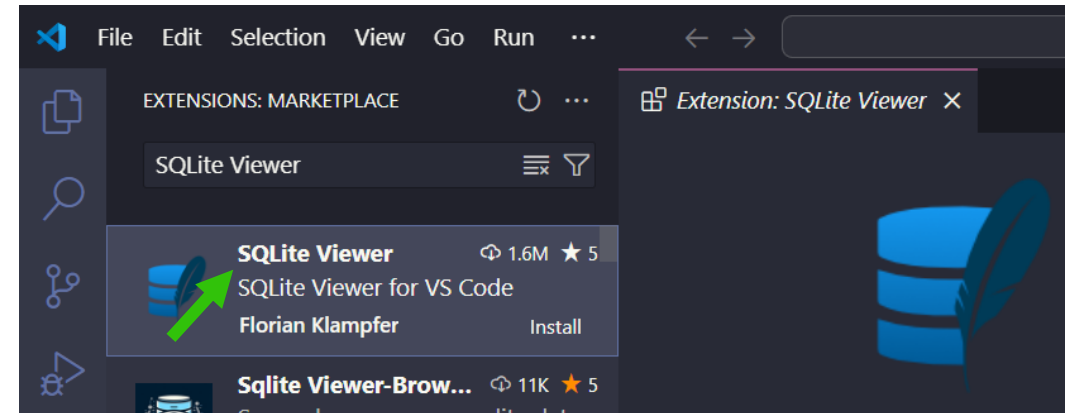
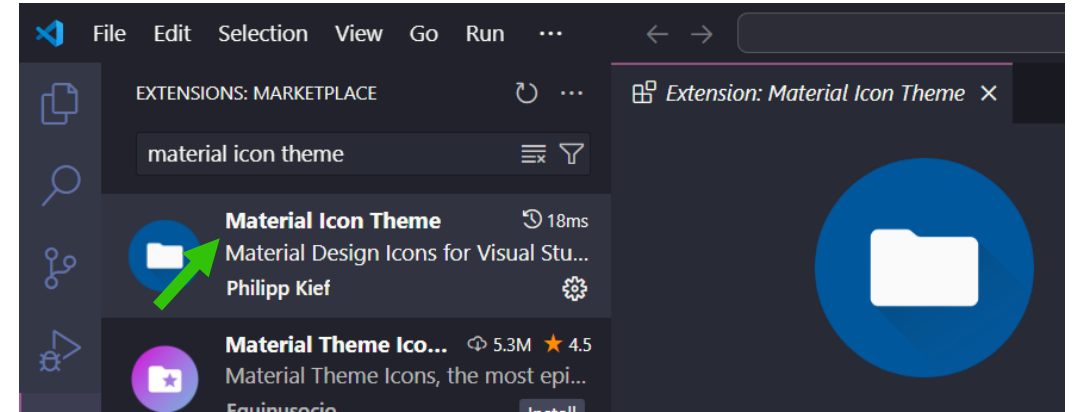
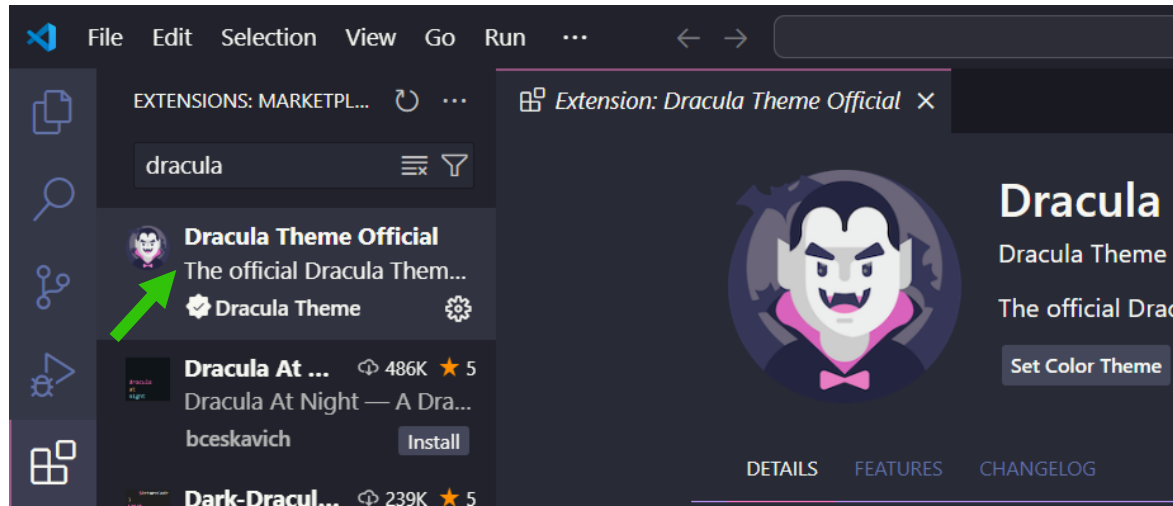
Your code editor. Redefined with AI.



[Download for Windows](#)

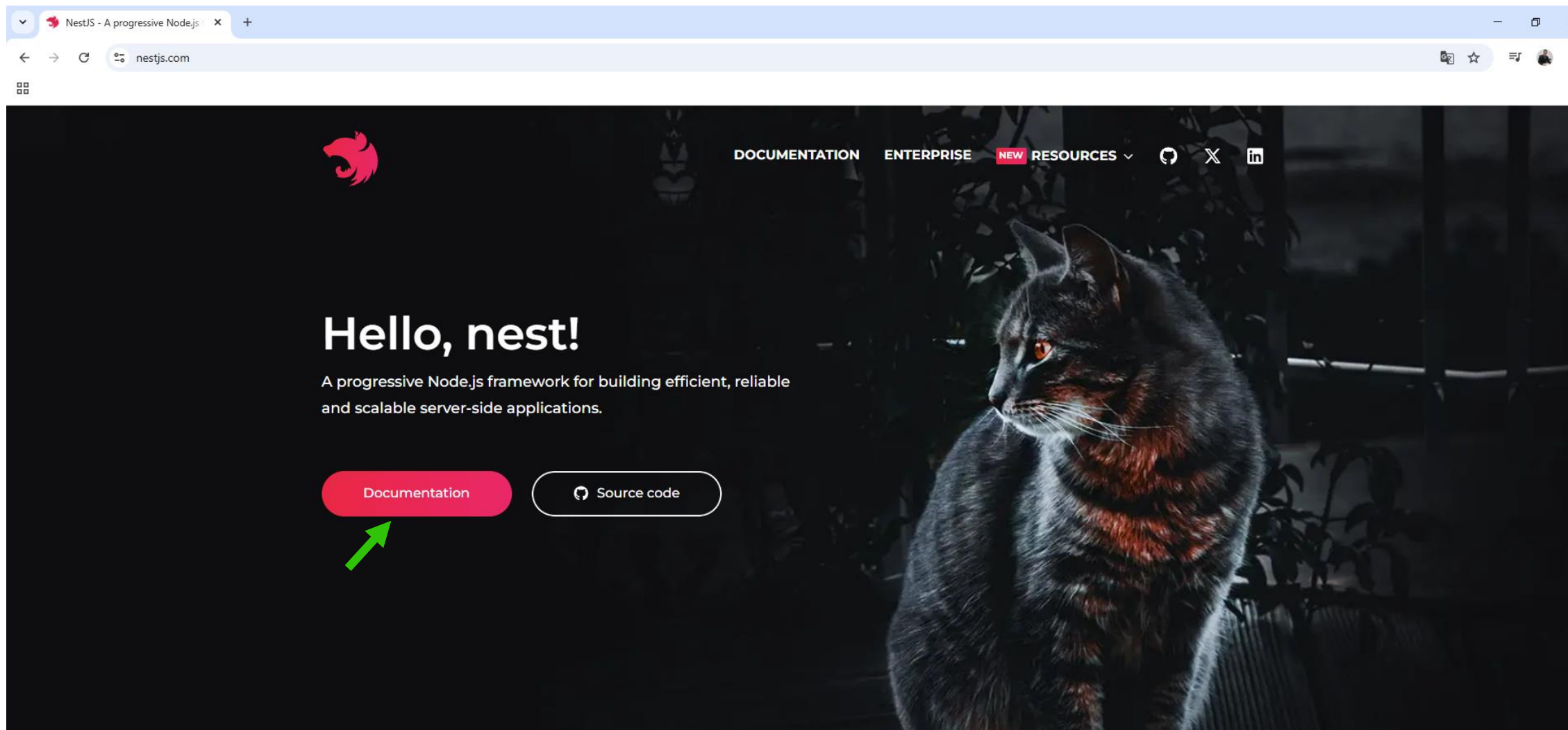
[Get Copilot Free](#)

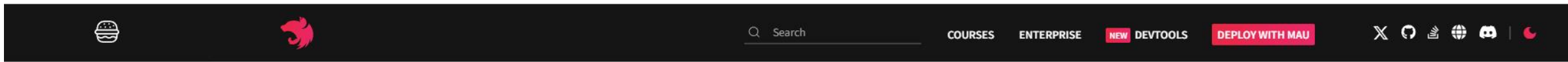
[Web](#), [Insiders edition](#), or [other platforms](#)





Nest JS





INTRODUCTION

OVERVIEW

- First steps
- Controllers
- Providers
- Modules
- Middleware
- Exception filters
- Pipes
- Guards
- Interceptors
- Custom decorators

FUNDAMENTALS

TECHNIQUES

SECURITY

GRAPHQL

WEBSOCKETS

MICROSERVICES

NEW DEPLOYMENT

STANDALONE APPS

CLI

OPENAPI

RECIPES

FAQ

DEVTOOLS

MIGRATION GUIDE

OFFICIAL COURSES

DISCOVER

Introduction

Nest (NestJS) is a framework for building efficient, scalable **Node.js** server-side applications. It uses progressive JavaScript, is built with and fully supports **TypeScript** (yet still enables developers to code in pure JavaScript) and combines elements of OOP (Object Oriented Programming), FP (Functional Programming), and FRP (Functional Reactive Programming).

Under the hood, Nest makes use of robust HTTP Server frameworks like **Express** (the default) and **Fastify** (an alternative). It also leverages the myriad of third-party modules which are available for the underlying platform.

Philosophy

In recent years, thanks to Node.js, JavaScript has become the “lingua franca” of the web for a wide range of technologies and **Vue**, which improve developer productivity and enable the creation of fast, testable, and maintainable applications. However, none of them effectively solve the main problem: how to structure a large application of many microservices, creating a conventional base structure for your project. Creating a new project with the **Nest CLI** is recommended for first-time users. We'll continue with this approach in **First Steps**.

Installation

To get started, you can either scaffold the project with the **Nest CLI**, or **clone a starter project**.

To scaffold the project with the Nest CLI, run the following commands. This will create a new project with the Nest CLI, creating a conventional base structure for your project. Creating a new project with the **Nest CLI** is recommended for first-time users. We'll continue with this approach in **First Steps**.

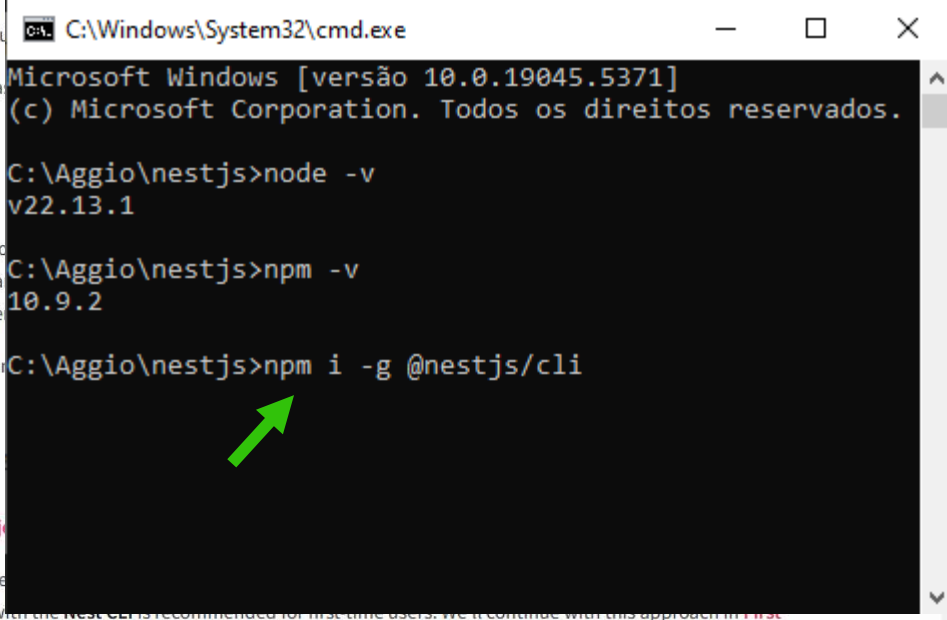
```
$ npm i -g @nestjs/cli
$ nest new project-name
```

Introduction

Philosophy

Installation

Alternatives



```
C:\Windows\system32\cmd.exe

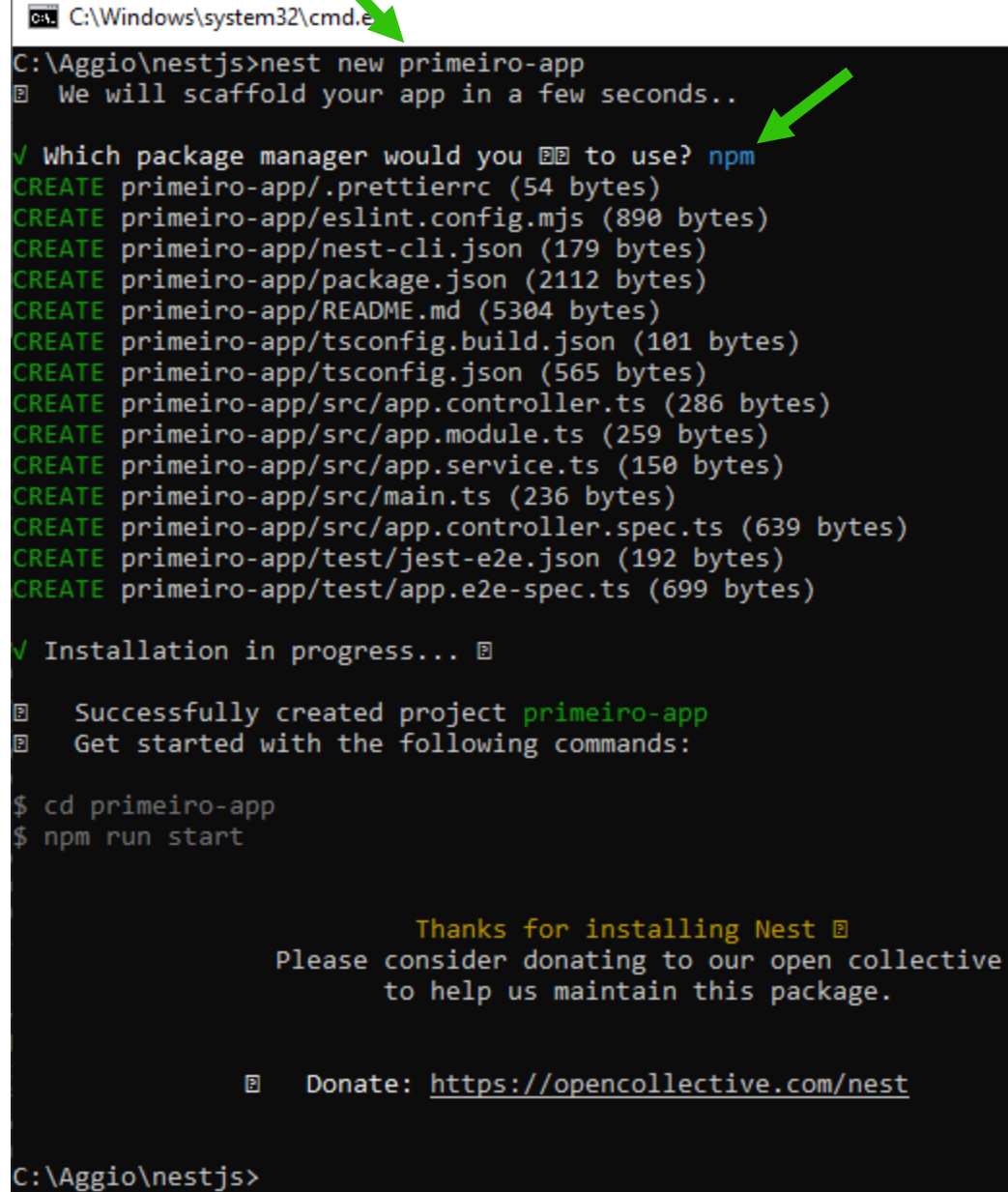
C:\Aggio\nestjs>nest --help
Usage: nest <command> [options]

Options:
  -v, --version          Output the current version.
  -h, --help              Output usage information.

Commands:
  new|n [options] [name]  Generate Nest application.
  build [options] [apps...] Build Nest application.
  start [options] [app]   Run Nest application.
  info|i                  Display Nest project details.
  add [options] <library> Adds support for an external library to your
                           project.
  generate|g [options] <schematic> [name] [path] Generate a Nest element.
                           Schematics available on @nestjs/schematics collection:
```

name	alias	description
application	application	Generate a new application workspace
class	cl	Generate a new class
configuration	config	Generate a CLI configuration file
controller	co	Generate a controller declaration
decorator	d	Generate a custom decorator
filter	f	Generate a filter declaration
gateway	ga	Generate a gateway declaration
guard	gu	Generate a guard declaration
interceptor	itc	Generate an interceptor declaration
interface	itf	Generate an interface
library	lib	Generate a new library within a monorepo
middleware	mi	Generate a middleware declaration
module	mo	Generate a module declaration
pipe	pi	Generate a pipe declaration
provider	pr	Generate a provider declaration
resolver	r	Generate a GraphQL resolver declaration
resource	res	Generate a new CRUD resource
service	s	Generate a service declaration
sub-app	app	Generate a new application within a monorepo

```
C:\Aggio\nestjs>
```



```
C:\Windows\system32\cmd.exe
C:\Aggio\nestjs>nest new primeiro-app
We will scaffold your app in a few seconds..

√ Which package manager would you like to use? npm
CREATE primeiro-app/.prettierrc (54 bytes)
CREATE primeiro-app/eslint.config.mjs (890 bytes)
CREATE primeiro-app/nest-cli.json (179 bytes)
CREATE primeiro-app/package.json (2112 bytes)
CREATE primeiro-app/README.md (5304 bytes)
CREATE primeiro-app/tsconfig.build.json (101 bytes)
CREATE primeiro-app/tsconfig.json (565 bytes)
CREATE primeiro-app/src/app.controller.ts (286 bytes)
CREATE primeiro-app/src/app.module.ts (259 bytes)
CREATE primeiro-app/src/app.service.ts (150 bytes)
CREATE primeiro-app/src/main.ts (236 bytes)
CREATE primeiro-app/src/app.controller.spec.ts (639 bytes)
CREATE primeiro-app/test/jest-e2e.json (192 bytes)
CREATE primeiro-app/test/app.e2e-spec.ts (699 bytes)

√ Installation in progress...

  Successfully created project primeiro-app
  Get started with the following commands:

$ cd primeiro-app
$ npm run start

      Thanks for installing Nest
    Please consider donating to our open collective
      to help us maintain this package.

  Donate: https://opencollective.com/nest

C:\Aggio\nestjs>
```

```
C:\Windows\system32\cmd.exe

✓ Installation in progress...

  Successfully created project primeiro-app
  Get started with the following commands:

$ cd primeiro-app
$ npm run start

      Thanks for installing Nest
  Please consider donating to our open collective
    to help us maintain this package.

  Donate: https://opencollective.com/nest

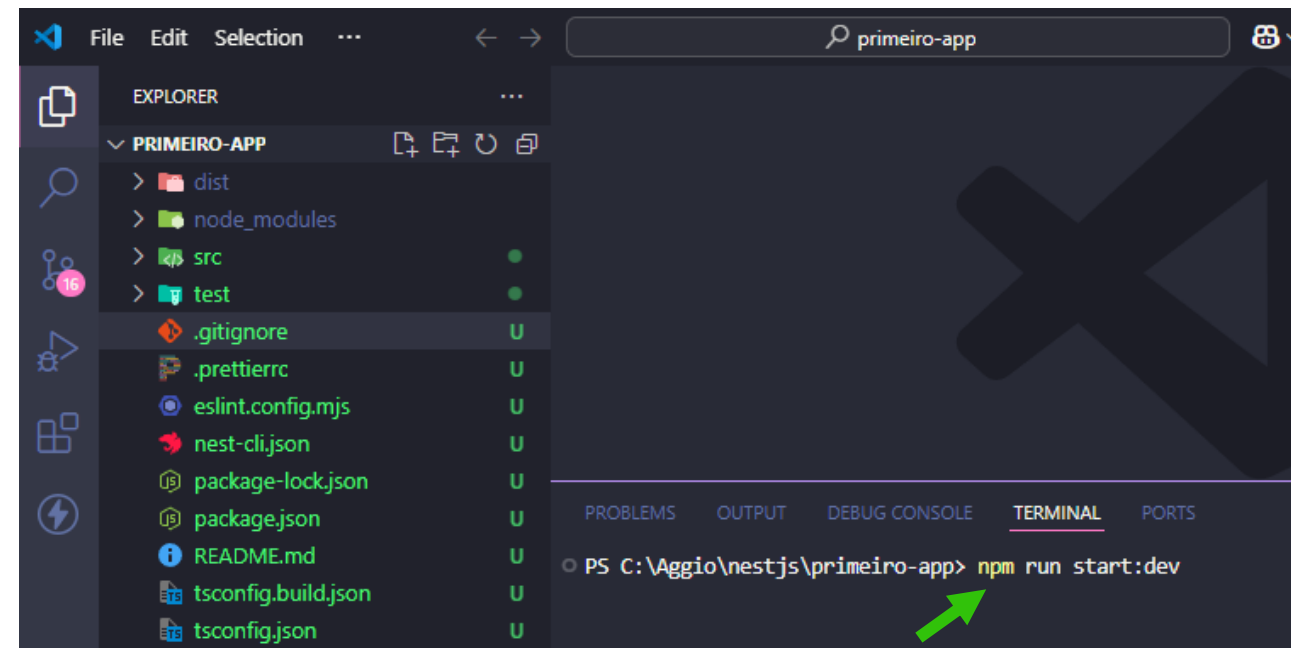
C:\Aggio\nestjs>cd primeiro-app
C:\Aggio\nestjs\primeiro-app>code .
C:\Aggio\nestjs\primeiro-app>
```

Terminal no VSCode:
ctrl + j

```
Prompt de Comando

C:\Aggio\nestjs>cd primeiro-app
C:\Aggio\nestjs\primeiro-app>code .
C:\Aggio\nestjs\primeiro-app>npm run start:dev
```

ou



```
File Edit Selection ... primeiro-app

EXPLORER
PRIMEIRO-APP
  > dist
  > node_modules
  > src
  > test
  > .gitignore
  > .prettierrc
  > eslint.config.mjs
  > nest-cli.json
  > package-lock.json
  > package.json
  > README.md
  > tsconfig.build.json
  > tsconfig.json

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Aggio\nestjs\primeiro-app> npm run start:dev
```

The image shows a web browser window at the top and a Visual Studio Code editor window at the bottom. The browser window has a single tab titled 'Documentation | NestJS - A pro...' and the address bar shows 'localhost:3000'. A green arrow points to the address bar. The browser displays 'Hello World!' with another green arrow pointing to the text. The VS Code editor window is titled 'primeiro-app' and shows the Explorer, Problems, Output, Debug Console, and Terminal panels. The Explorer panel shows the file structure of the 'PRIMEIRO-APP' project, including 'dist', 'node_modules', 'src', 'test', and various configuration files. The Terminal panel shows the output of the application, including logs for starting the Nest application and its dependencies.

Browser Address Bar: localhost:3000

Browser Content: Hello World!

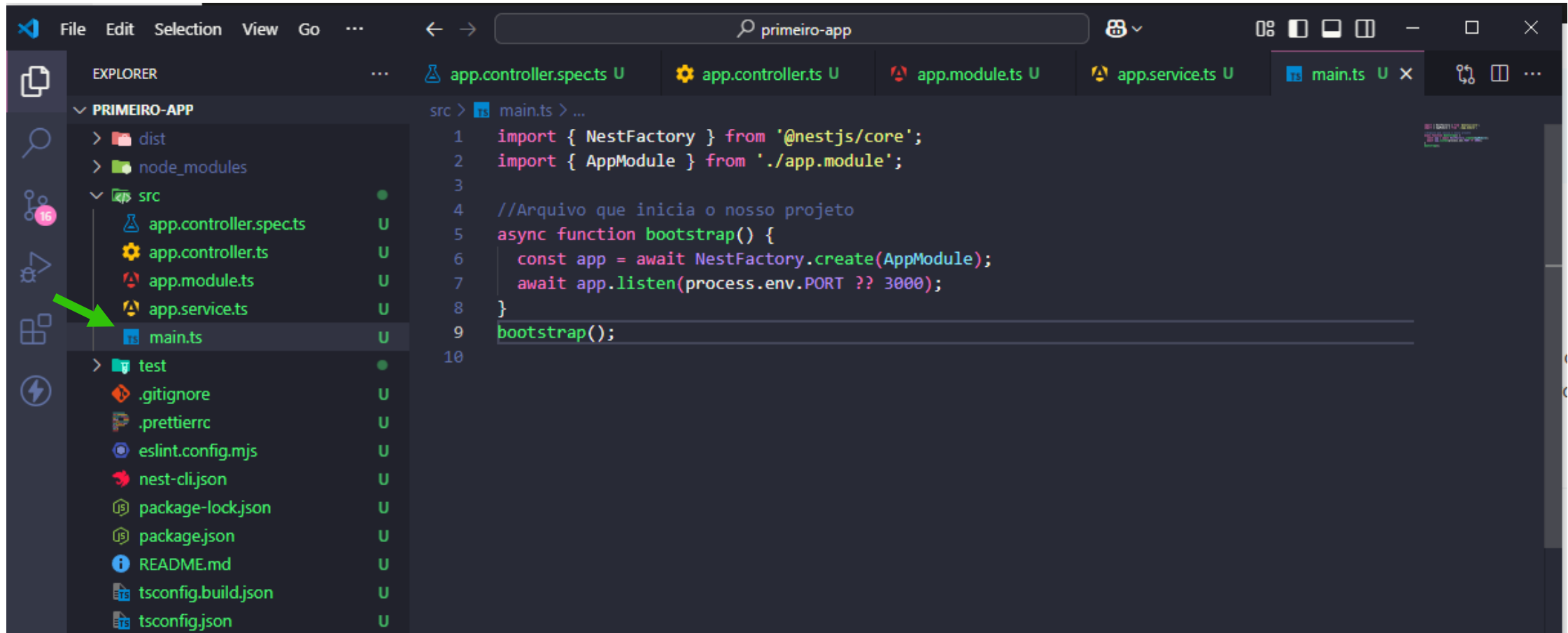
VS Code Explorer:

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.build.json
 - tsconfig.json

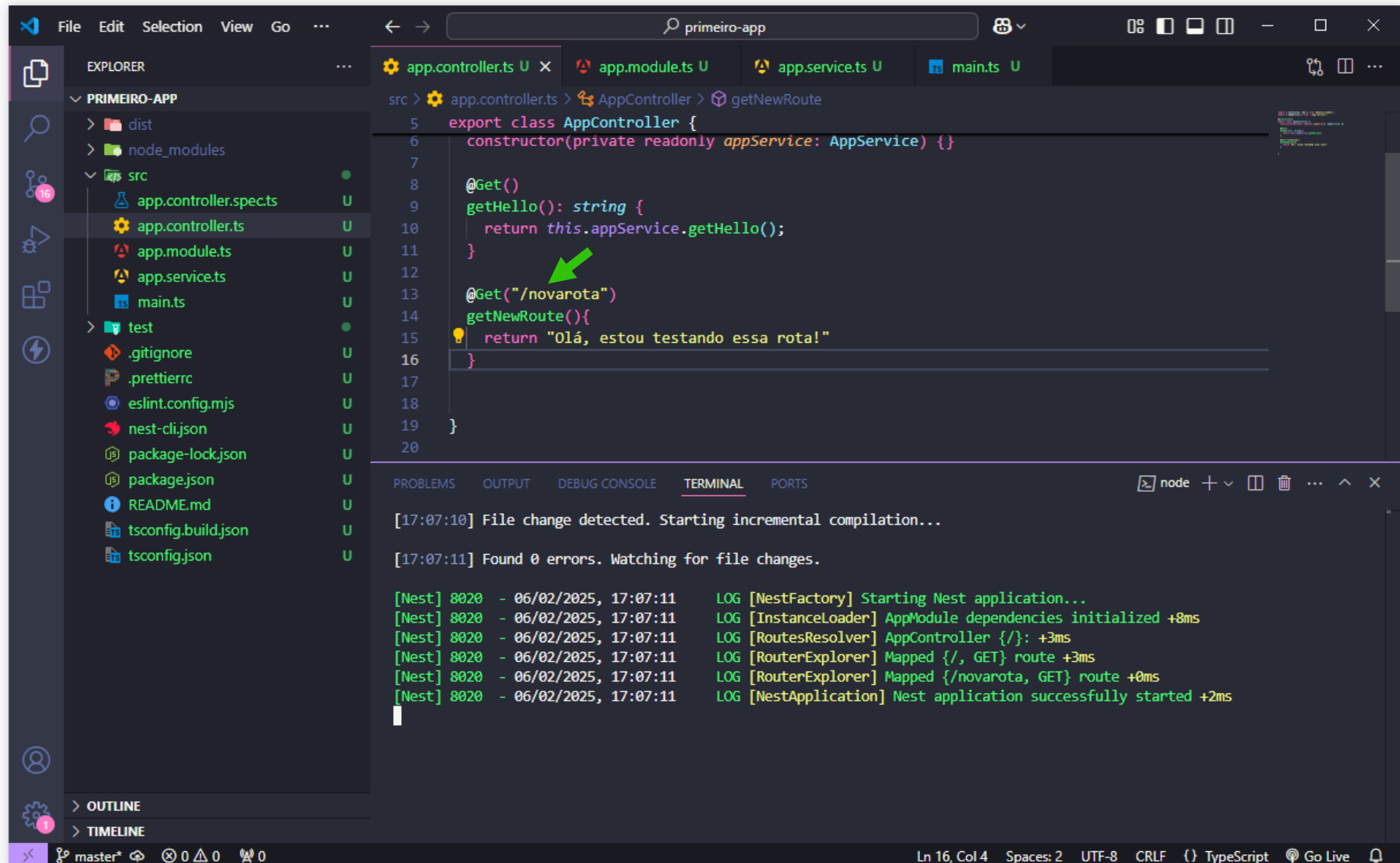
VS Code Terminal:

```
[16:34:50] Starting compilation in watch mode...
[16:34:52] Found 0 errors. Watching for file changes.
[16:34:50] Starting compilation in watch mode...
[16:34:52] Found 0 errors. Watching for file changes.
[Nest] 16336 - 06/02/2025, 16:34:53 LOG [NestFactory] Starting Nest applicatio
[16:34:50] Starting compilation in watch mode...
[16:34:52] Found 0 errors. Watching for file changes.
[Nest] 16336 - 06/02/2025, 16:34:53 LOG [NestFactory] Starting Nest applicatio
[16:34:50] Starting compilation in watch mode...
[16:34:52] Found 0 errors. Watching for file changes.
[16:34:50] Starting compilation in watch mode...
[16:34:50] Starting compilation in watch mode...
[16:34:50] Starting compilation in watch mode...
[16:34:52] Found 0 errors. Watching for file changes.
[Nest] 16336 - 06/02/2025, 16:34:53 LOG [NestFactory] Starting Nest application...
[Nest] 16336 - 06/02/2025, 16:34:53 LOG [InstanceLoader] AppModule dependencies initialized +8ms
[Nest] 16336 - 06/02/2025, 16:34:53 LOG [RoutesResolver] AppController {/}: +4ms
[Nest] 16336 - 06/02/2025, 16:34:53 LOG [RouterExplorer] Mapped {/, GET} route +2ms
[Nest] 16336 - 06/02/2025, 16:34:53 LOG [NestApplication] Nest application successfully started +2ms
```

Estrutura do Projeto



Olá, estou testando essa rota!



The image shows a Visual Studio Code editor window with a project named "primeiro-app". The Explorer sidebar on the left shows the file structure, including a "src" directory with files like "app.controller.spec.ts", "app.controller.ts", "app.module.ts", "app.service.ts", and "main.ts". The main editor area displays the code for "app.controller.ts", which defines an "AppController" class with a "getNewRoute" method. A green arrow points to the "@Get('/novarota')" decorator in the "getNewRoute" method. The terminal at the bottom shows the output of the application, including logs for the NestJS framework and the successful start of the application.

```
src > app.controller.ts > AppController > getNewRoute
5 export class AppController {
6   constructor(private readonly appService: AppService) {}
7
8   @Get()
9   getHello(): string {
10     return this.appService.getHello();
11   }
12
13   @Get("/novarota")
14   getNewRoute(){
15     return "Olá, estou testando essa rota!"
16   }
17
18 }
19
20
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

[17:07:10] File change detected. Starting incremental compilation...

[17:07:11] Found 0 errors. Watching for file changes.

[Nest] 8020 - 06/02/2025, 17:07:11 LOG [NestFactory] Starting Nest application...

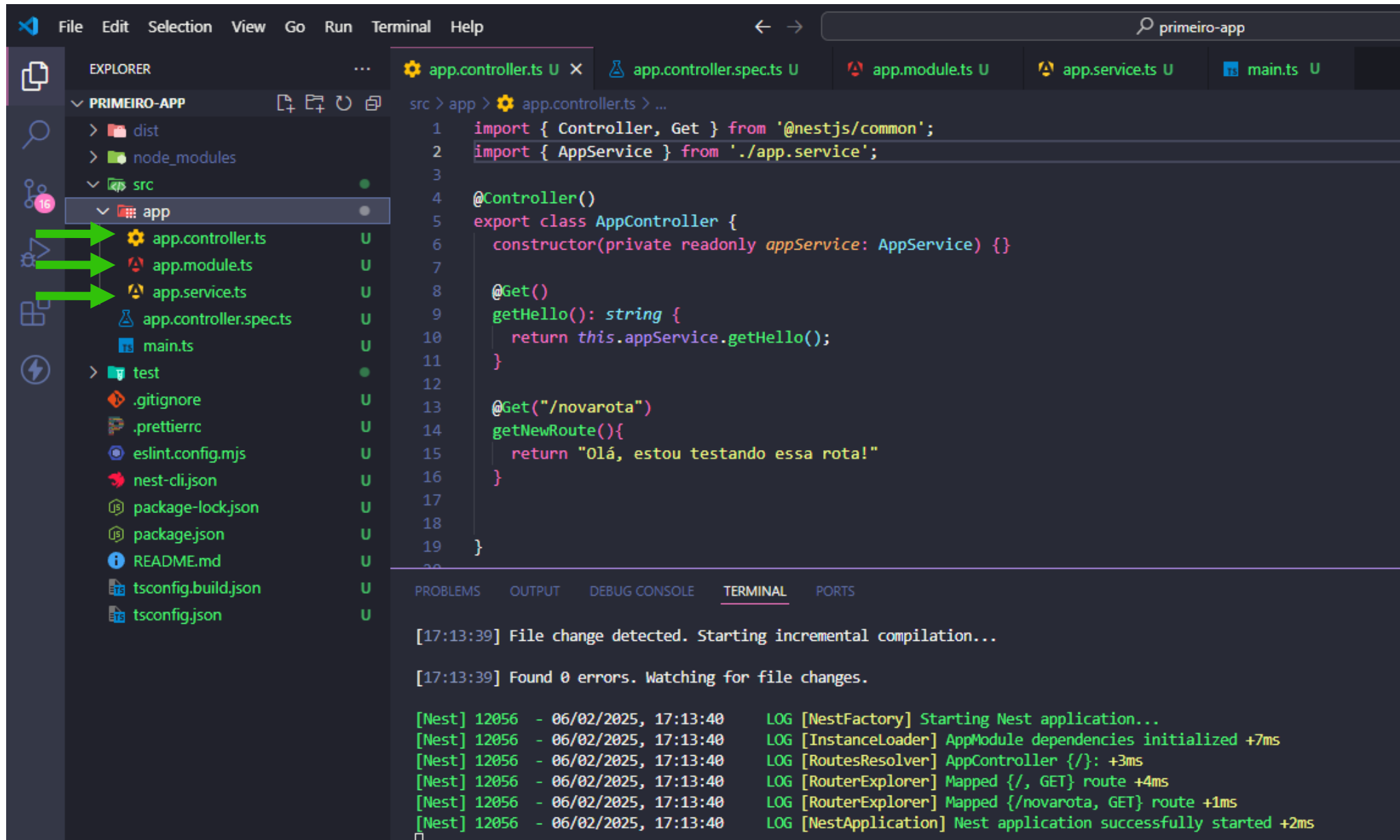
[Nest] 8020 - 06/02/2025, 17:07:11 LOG [InstanceLoader] AppModule dependencies initialized +8ms

[Nest] 8020 - 06/02/2025, 17:07:11 LOG [RoutesResolver] AppController {/}: +3ms

[Nest] 8020 - 06/02/2025, 17:07:11 LOG [RouterExplorer] Mapped {/, GET} route +3ms

[Nest] 8020 - 06/02/2025, 17:07:11 LOG [RouterExplorer] Mapped {/novarota, GET} route +0ms

[Nest] 8020 - 06/02/2025, 17:07:11 LOG [NestApplication] Nest application successfully started +2ms



File Edit Selection View Go Run Terminal Help

primeiro-app

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - app.controller.ts
 - app.module.ts
 - app.service.ts
 - app.controller.spec.ts
 - main.ts
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.build.json
 - tsconfig.json

src > app > app.controller.ts > ...

```

1 import { Controller, Get } from '@nestjs/common';
2 import { AppService } from './app.service';
3
4 @Controller()
5 export class AppController {
6   constructor(private readonly appService: AppService) {}
7
8   @Get()
9   getHello(): string {
10     return this.appService.getHello();
11   }
12
13   @Get("/novarota")
14   getNewRoute(){
15     return "Olá, estou testando essa rota!"
16   }
17
18 }
19

```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

```

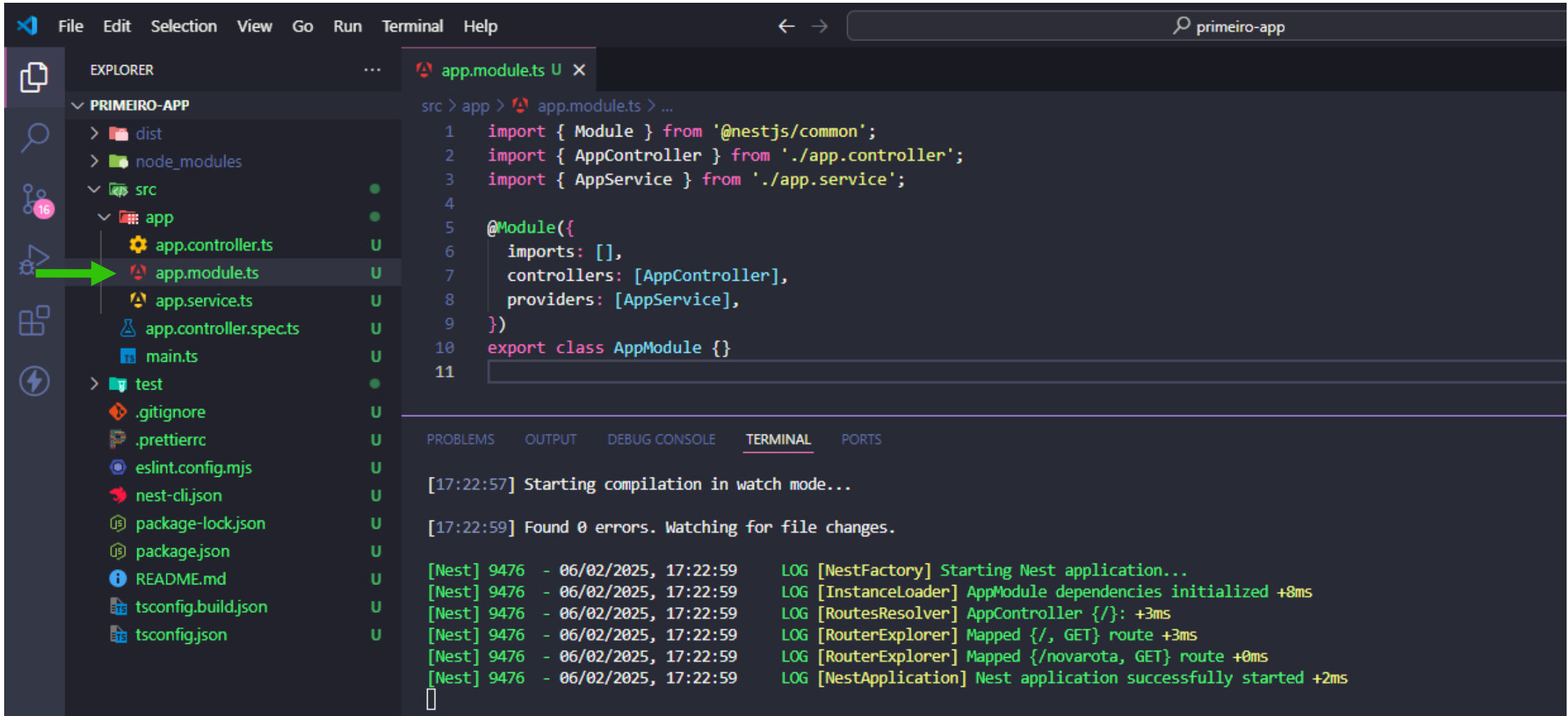
[17:13:39] File change detected. Starting incremental compilation...

[17:13:39] Found 0 errors. Watching for file changes.

[Nest] 12056 - 06/02/2025, 17:13:40 LOG [NestFactory] Starting Nest application...
[Nest] 12056 - 06/02/2025, 17:13:40 LOG [InstanceLoader] AppModule dependencies initialized +7ms
[Nest] 12056 - 06/02/2025, 17:13:40 LOG [RoutesResolver] AppController {/}: +3ms
[Nest] 12056 - 06/02/2025, 17:13:40 LOG [RouterExplorer] Mapped {/, GET} route +4ms
[Nest] 12056 - 06/02/2025, 17:13:40 LOG [RouterExplorer] Mapped {/novarota, GET} route +1ms
[Nest] 12056 - 06/02/2025, 17:13:40 LOG [NestApplication] Nest application successfully started +2ms

```

Módulo Principal do Projeto



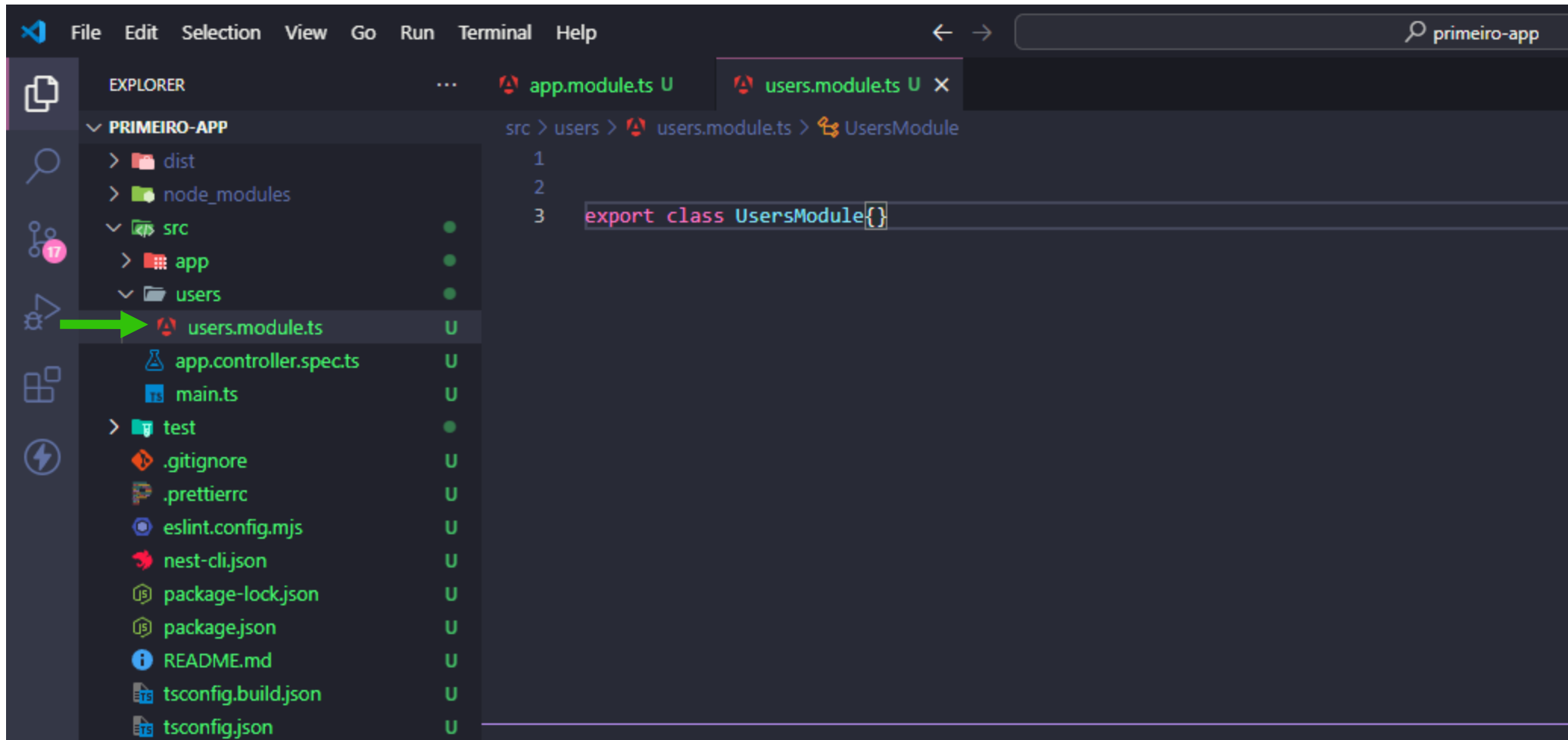
The screenshot displays the Visual Studio Code interface for a project named 'primeiro-app'. The Explorer sidebar on the left shows the project structure, with a green arrow pointing to 'app.module.ts' under the 'app' directory. The main editor shows the content of 'app.module.ts'.

```
src > app > app.module.ts > ...
1  import { Module } from '@nestjs/common';
2  import { AppController } from './app.controller';
3  import { AppService } from './app.service';
4
5  @Module({
6    imports: [],
7    controllers: [AppController],
8    providers: [AppService],
9  })
10 export class AppModule {}
11
```

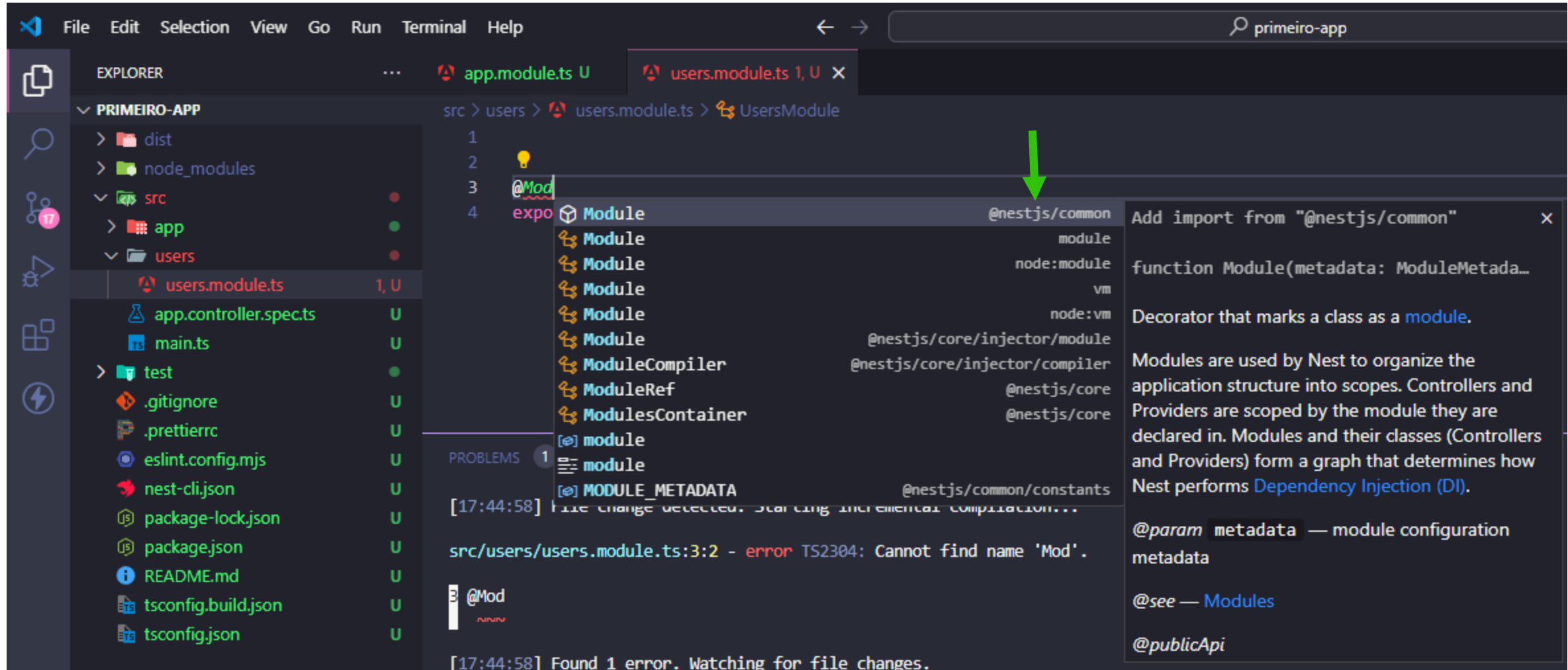
The bottom panel shows the TERMINAL output, indicating that the application is running successfully.

```
[17:22:57] Starting compilation in watch mode...
[17:22:59] Found 0 errors. Watching for file changes.
[Nest] 9476 - 06/02/2025, 17:22:59 LOG [NestFactory] Starting Nest application...
[Nest] 9476 - 06/02/2025, 17:22:59 LOG [InstanceLoader] AppModule dependencies initialized +8ms
[Nest] 9476 - 06/02/2025, 17:22:59 LOG [RoutesResolver] AppController {/}: +3ms
[Nest] 9476 - 06/02/2025, 17:22:59 LOG [RouterExplorer] Mapped {/, GET} route +3ms
[Nest] 9476 - 06/02/2025, 17:22:59 LOG [RouterExplorer] Mapped {/novarota, GET} route +0ms
[Nest] 9476 - 06/02/2025, 17:22:59 LOG [NestApplication] Nest application successfully started +2ms
```

Criando um Módulo



Criando um Módulo



File Edit Selection View Go Run Terminal Help

primeiro-app

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - users.module.ts 1, U

src > users > users.module.ts > UsersModule

```

1
2
3 @Mod
4 expo

```

Module @nestjs/common

Module module

Module node:module

Module vm

Module node:vm

Module @nestjs/core/injector/module

ModuleCompiler @nestjs/core/injector/compiler

ModuleRef @nestjs/core

ModulesContainer @nestjs/core

module

module

MODULE_METADATA @nestjs/common/constants

[17:44:58] File change detected. Starting incremental compilation...

src/users/users.module.ts:3:2 - error TS2304: Cannot find name 'Mod'.

@Mod

[17:44:58] Found 1 error. Watching for file changes.

Add import from "@nestjs/common"

function Module(metadata: ModuleMetada...

Decorator that marks a class as a **module**.

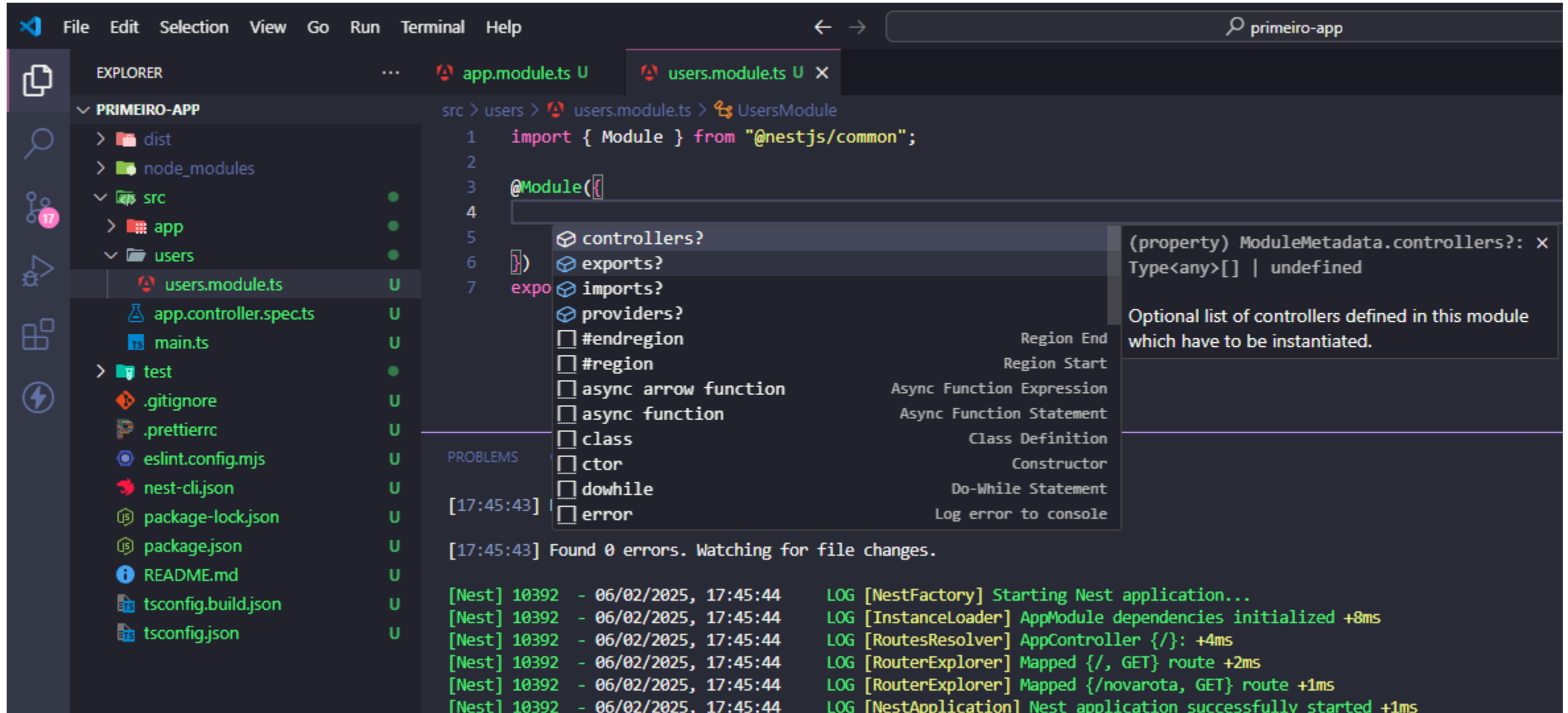
Modules are used by Nest to organize the application structure into scopes. Controllers and Providers are scoped by the module they are declared in. Modules and their classes (Controllers and Providers) form a graph that determines how Nest performs **Dependency Injection (DI)**.

@param metadata — module configuration metadata

@see — [Modules](#)

@publicApi

Criando um Módulo



The screenshot shows the Visual Studio Code interface with the Explorer, Source Control, and Run and Debug views. The Explorer view shows the project structure for 'primeiro-app', including the 'src' directory and the 'users' module. The Source Control view shows the 'users.module.ts' file. The Run and Debug view shows the output of the application, including the 'Found 0 errors. Watching for file changes.' message and the 'Starting Nest application...' log.

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - users
 - users.module.ts
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.build.json
 - tsconfig.json

users.module.ts

```
1 import { Module } from "@nestjs/common";
2
3 @Module({
4
5   controllers?
6   exports?
7   imports?
8   providers?
9   #endregion
10  #region
11  async arrow function
12  async function
13  class
14  ctor
15  dowhile
16  error
17 })
18 export class UsersModule {}
```

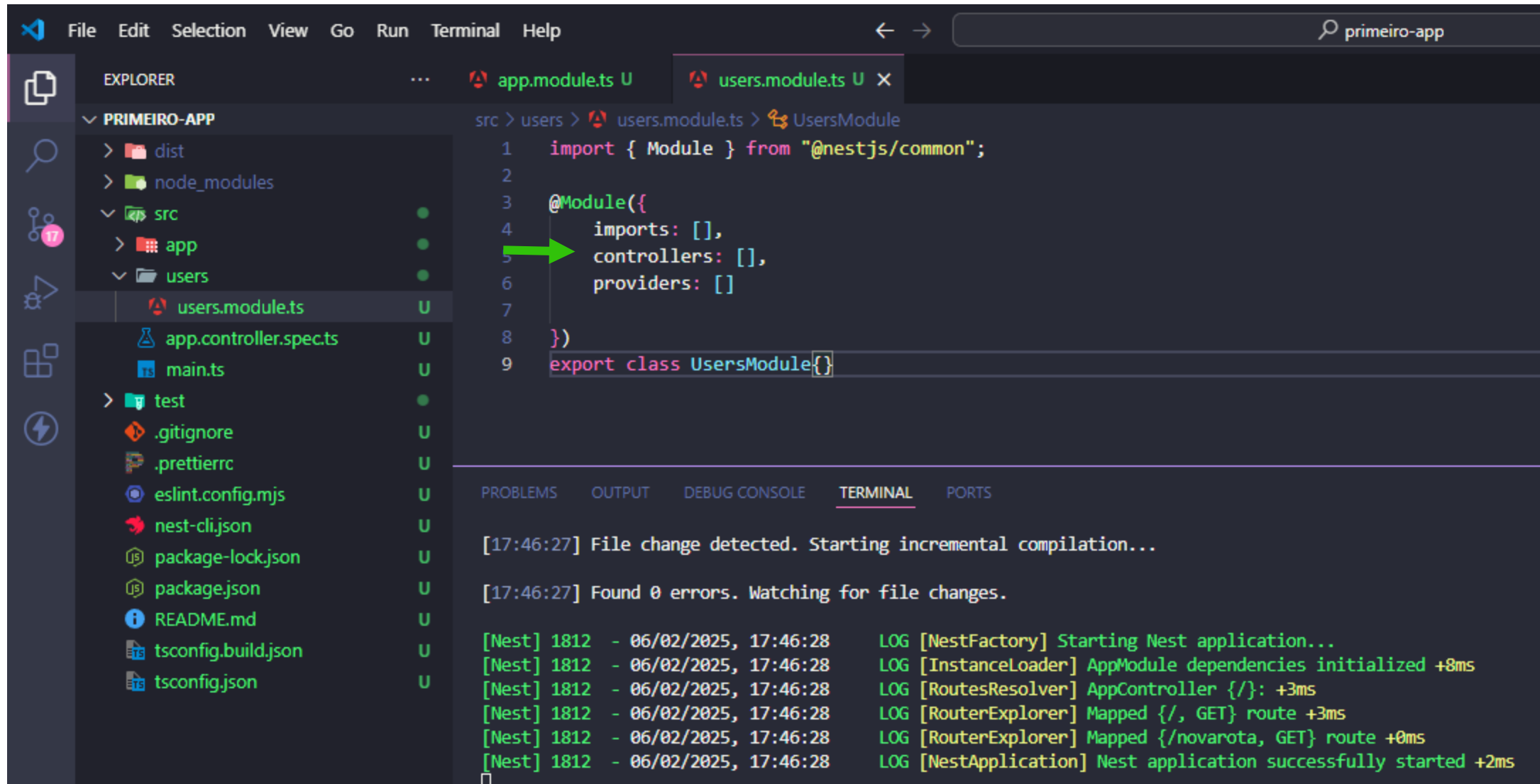
PROBLEMS

[17:45:43] Found 0 errors. Watching for file changes.

OUTPUT

```
[Nest] 10392 - 06/02/2025, 17:45:44 LOG [NestFactory] Starting Nest application...
[Nest] 10392 - 06/02/2025, 17:45:44 LOG [InstanceLoader] AppModule dependencies initialized +8ms
[Nest] 10392 - 06/02/2025, 17:45:44 LOG [RoutesResolver] AppController {/}: +4ms
[Nest] 10392 - 06/02/2025, 17:45:44 LOG [RouterExplorer] Mapped {/, GET} route +2ms
[Nest] 10392 - 06/02/2025, 17:45:44 LOG [RouterExplorer] Mapped {/novarota, GET} route +1ms
[Nest] 10392 - 06/02/2025, 17:45:44 LOG [NestApplication] Nest application successfully started +1ms
```

Criando um Módulo



The screenshot shows the Visual Studio Code interface with a project named "PRIMEIRO-APP". The Explorer sidebar on the left shows the file structure, including a "users" folder. The main editor displays the "users.module.ts" file, which contains the following code:

```
1 import { Module } from "@nestjs/common";
2
3 @Module({
4   imports: [],
5   controllers: [],
6   providers: []
7 })
8
9 export class UsersModule {}
```

A green arrow points to the "controllers: []" property in the `@Module` decorator. The bottom panel shows the "TERMINAL" output, which includes the following messages:

```
[17:46:27] File change detected. Starting incremental compilation...
[17:46:27] Found 0 errors. Watching for file changes.
[Nest] 1812 - 06/02/2025, 17:46:28 LOG [NestFactory] Starting Nest application...
[Nest] 1812 - 06/02/2025, 17:46:28 LOG [InstanceLoader] AppModule dependencies initialized +8ms
[Nest] 1812 - 06/02/2025, 17:46:28 LOG [RoutesResolver] AppController {/}: +3ms
[Nest] 1812 - 06/02/2025, 17:46:28 LOG [RouterExplorer] Mapped {/, GET} route +3ms
[Nest] 1812 - 06/02/2025, 17:46:28 LOG [RouterExplorer] Mapped {/novarota, GET} route +0ms
[Nest] 1812 - 06/02/2025, 17:46:28 LOG [NestApplication] Nest application successfully started +2ms
```

Criando um Módulo

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

[17:47:53] Starting compilation in watch mode...

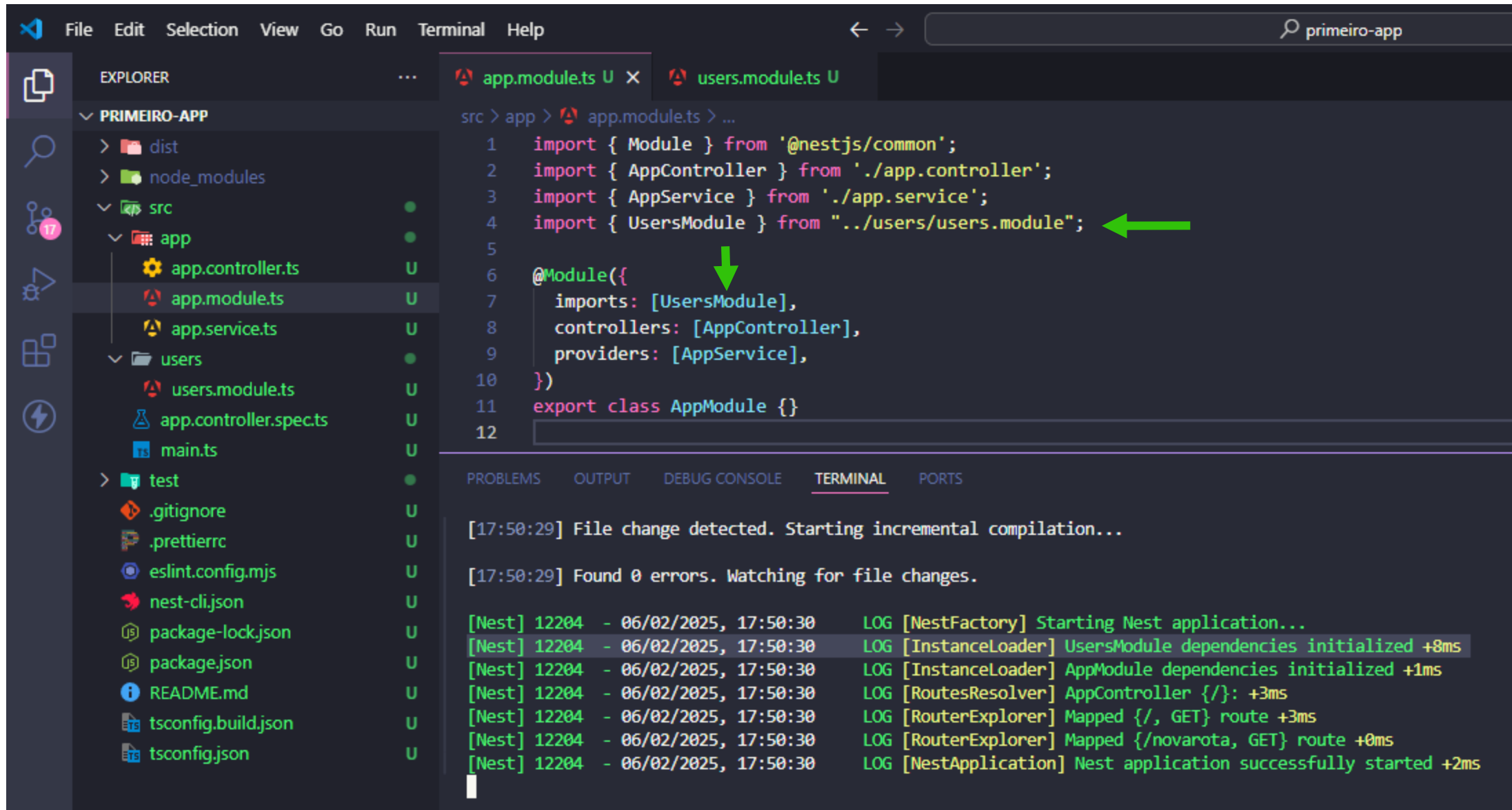
[17:47:55] Found 0 errors. Watching for file changes.

[Nest] 13672 - 06/02/2025, 17:47:56 LOG [NestFactory] Starting Nest application...
[Nest] 13672 - 06/02/2025, 17:47:56 LOG [InstanceLoader] AppModule dependencies initialized +7ms
[Nest] 13672 - 06/02/2025, 17:47:56 LOG [RoutesResolver] AppController {/}: +4ms
[Nest] 13672 - 06/02/2025, 17:47:56 LOG [RouterExplorer] Mapped {/, GET} route +2ms
[Nest] 13672 - 06/02/2025, 17:47:56 LOG [RouterExplorer] Mapped {/novarota, GET} route +2ms
[Nest] 13672 - 06/02/2025, 17:47:56 LOG [NestApplication] Nest application successfully started +2ms
```

Precisa importar o módulo users



Criando um Módulo



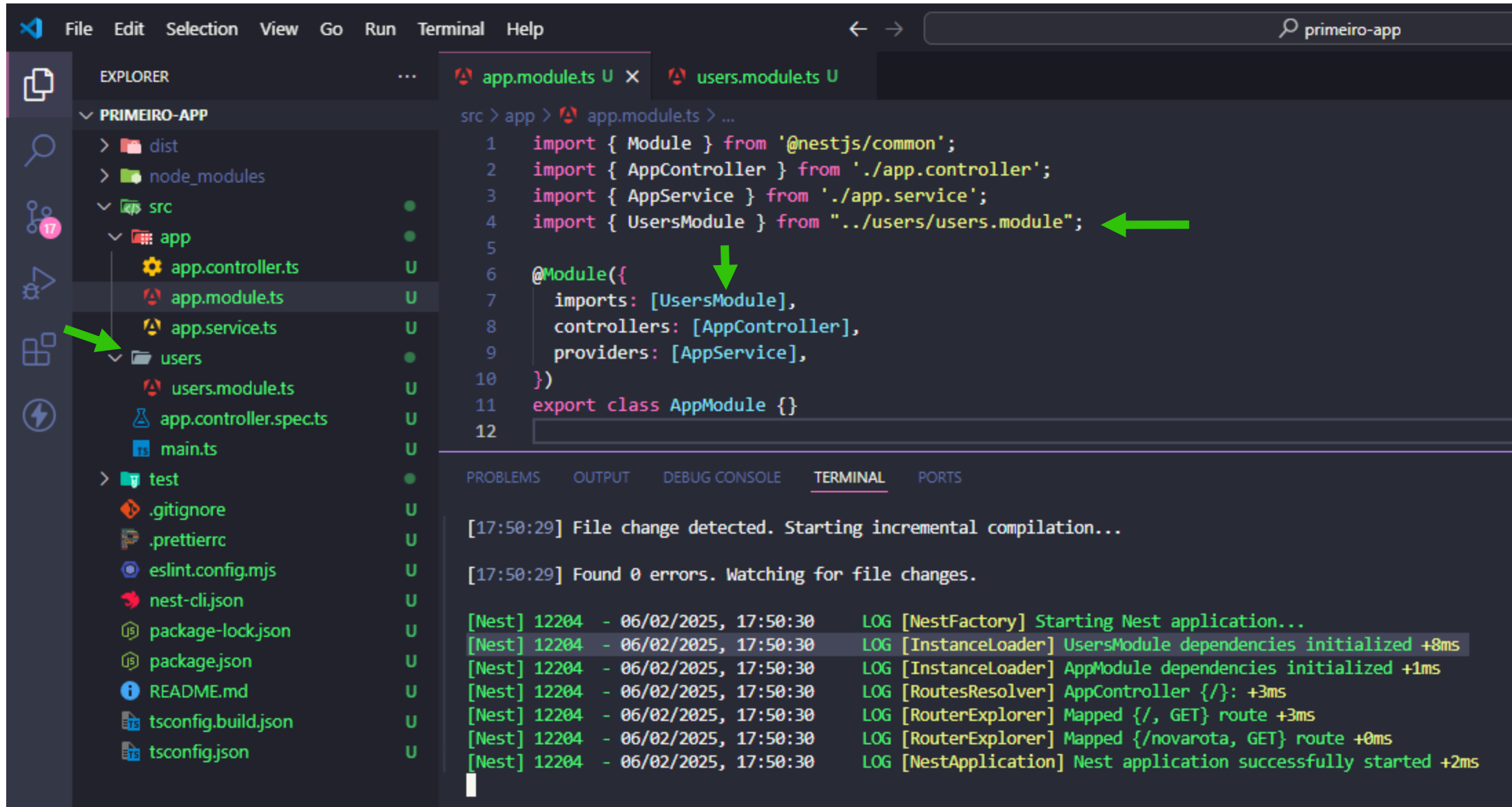
The screenshot displays the Visual Studio Code interface for a NestJS application. The Explorer on the left shows the project structure, with the 'app' folder expanded and 'app.module.ts' selected. The Editor on the right shows the code for 'app.module.ts'.

```
src > app > app.module.ts > ...
1  import { Module } from '@nestjs/common';
2  import { AppController } from './app.controller';
3  import { AppService } from './app.service';
4  import { UsersModule } from '../users/users.module';
5
6  @Module({
7    imports: [UsersModule],
8    controllers: [AppController],
9    providers: [AppService],
10 })
11 export class AppModule {}
12
```

The Terminal at the bottom shows the output of the application startup, including logs for module dependencies and successful startup.

```
[17:50:29] File change detected. Starting incremental compilation...
[17:50:29] Found 0 errors. Watching for file changes.
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [NestFactory] Starting Nest application...
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [InstanceLoader] UsersModule dependencies initialized +8ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [InstanceLoader] AppModule dependencies initialized +1ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [RoutesResolver] AppController {/}: +3ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [RouterExplorer] Mapped {/, GET} route +3ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [RouterExplorer] Mapped {/novarota, GET} route +0ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [NestApplication] Nest application successfully started +2ms
```

Excluir as setas verdes



The image shows a Visual Studio Code editor with a NestJS application. The Explorer panel on the left displays the file structure of a project named 'PRIMEIRO-APP'. The main editor shows the 'app.module.ts' file, which contains imports for '@nestjs/common', 'AppController', 'AppService', and 'UsersModule'. A green arrow points to the 'UsersModule' import, and another points to the 'users' folder in the Explorer. The terminal at the bottom shows the application starting successfully.

```

src > app > app.module.ts > ...
1  import { Module } from '@nestjs/common';
2  import { AppController } from './app.controller';
3  import { AppService } from './app.service';
4  import { UsersModule } from '../users/users.module';
5
6  @Module({
7    imports: [UsersModule],
8    controllers: [AppController],
9    providers: [AppService],
10 })
11 export class AppModule {}
12

```

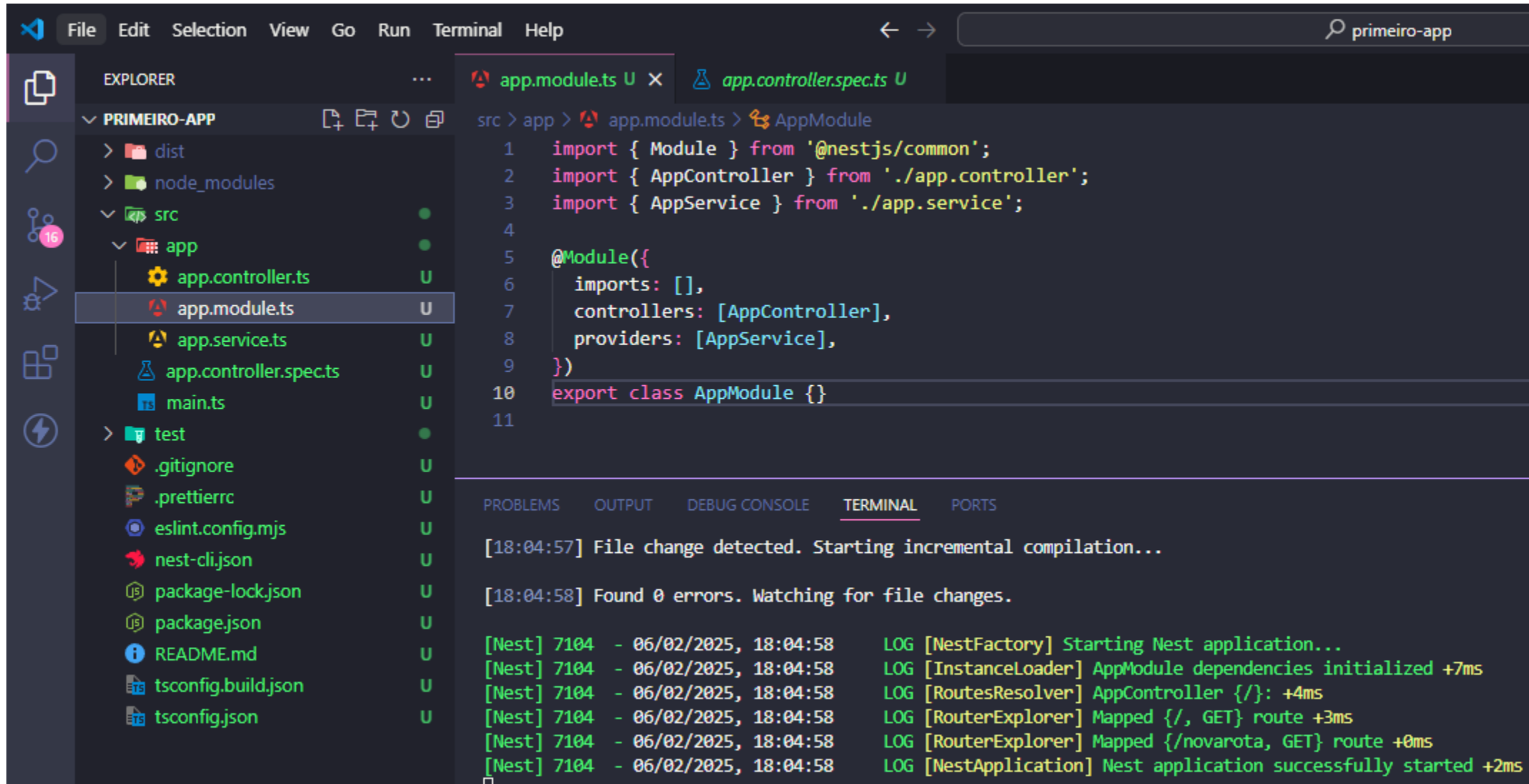
Terminal output:

```

[17:50:29] File change detected. Starting incremental compilation...
[17:50:29] Found 0 errors. Watching for file changes.
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [NestFactory] Starting Nest application...
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [InstanceLoader] UsersModule dependencies initialized +8ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [InstanceLoader] AppModule dependencies initialized +1ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [RoutesResolver] AppController {/}: +3ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [RouterExplorer] Mapped {/, GET} route +3ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [RouterExplorer] Mapped {/novarota, GET} route +0ms
[Nest] 12204 - 06/02/2025, 17:50:30 LOG [NestApplication] Nest application successfully started +2ms

```

Criando um Módulo



The screenshot shows the Visual Studio Code interface with the Explorer, Source Control, and Run and Debug views. The Explorer view shows the project structure for 'PRIMEIRO-APP', including the 'src' directory and the 'app' module. The 'app' module is expanded, showing files like 'app.controller.ts', 'app.module.ts', 'app.service.ts', 'app.controller.spec.ts', and 'main.ts'. The 'app.module.ts' file is selected, and its content is displayed in the editor. The code defines the 'AppModule' class, which imports 'AppController' and 'AppService' and registers them as controllers and providers. The terminal view shows the output of the application, including the message 'Nest application successfully started'.

```
File Edit Selection View Go Run Terminal Help
```

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - app.controller.ts
 - app.module.ts
 - app.service.ts
 - app.controller.spec.ts
 - main.ts
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.build.json
 - tsconfig.json

src > app > app.module.ts > AppModule

```
1 import { Module } from '@nestjs/common';
2 import { AppController } from './app.controller';
3 import { AppService } from './app.service';
4
5 @Module({
6   imports: [],
7   controllers: [AppController],
8   providers: [AppService],
9 })
10 export class AppModule {}
11
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[18:04:57] File change detected. Starting incremental compilation...
[18:04:58] Found 0 errors. Watching for file changes.
[Nest] 7104 - 06/02/2025, 18:04:58 LOG [NestFactory] Starting Nest application...
[Nest] 7104 - 06/02/2025, 18:04:58 LOG [InstanceLoader] AppModule dependencies initialized +7ms
[Nest] 7104 - 06/02/2025, 18:04:58 LOG [RoutesResolver] AppController {/}: +4ms
[Nest] 7104 - 06/02/2025, 18:04:58 LOG [RouterExplorer] Mapped {/, GET} route +3ms
[Nest] 7104 - 06/02/2025, 18:04:58 LOG [RouterExplorer] Mapped {/novarota, GET} route +0ms
[Nest] 7104 - 06/02/2025, 18:04:58 LOG [NestApplication] Nest application successfully started +2ms
```

Criando um Módulo com CLI

Visual Studio Code interface showing the Explorer, Editor, and Terminal panels.

EXPLORER: The file explorer on the left shows the project structure for 'PRIMEIRO-APP'. The 'app' folder is expanded, showing files like 'app.controller.ts', 'app.module.ts', 'app.service.ts', 'app.controller.spec.ts', 'main.ts', and 'test'. A green arrow points to 'app.module.ts'.

EDITOR: The editor shows the content of 'app.module.ts' with the following code:

```
1 import { Module } from '@nestjs/common';
2 import { AppController } from './app.controller';
3 import { AppService } from './app.service';
4
```

TERMINAL: The terminal panel at the bottom shows the command 'nest --help' and its output:

```
PS C:\Aggio\nestjs\primeiro-app> nest --help
Usage: nest <command> [options]

Options:
  -v, --version           Output the current version.
  -h, --help              Output usage information.

Commands:
  new[n [options] [name]] Generate Nest application.
  build [options] [apps...] Build Nest application.
  start [options] [app]    Run Nest application.
  info[i]                  Display Nest project details.
  add [options] <library>  Adds support for an external library to your project.
  generate[g [options] <schematic> [name] [path]] Generate a Nest element.
                           Schematics available on @nestjs/schematics collection:
```

A green arrow points to the 'generate' command. Below the terminal output, a table lists the available schematics:

name	alias	description
application	application	Generate a new application workspace
class	cl	Generate a new class
configuration	config	Generate a CLI configuration file
controller	co	Generate a controller declaration
decorator	d	Generate a custom decorator
filter	f	Generate a filter declaration
gateway	ga	Generate a gateway declaration
guard	gu	Generate a guard declaration
interceptor	itc	Generate an interceptor declaration
interface	itf	Generate an interface
library	lib	Generate a new library within a monorepo
middleware	mi	Generate a middleware declaration
module	mo	Generate a module declaration
pipe	pi	Generate a pipe declaration
provider	pr	Generate a provider declaration
resolver	r	Generate a GraphQL resolver declaration
resource	res	Generate a new CRUD resource
service	s	Generate a service declaration
sub-app	app	Generate a new application within a monorepo

At the bottom of the terminal, the command 'nest g module tasks' is entered, with a green arrow pointing to it.

Criando um Módulo com CLI

Visual Studio Code interface showing the creation of a new module in a NestJS application.

EXPLORER: The file explorer on the left shows the project structure. The `tasks` folder is expanded, and `tasks.module.ts` is selected.

EDITOR: The editor shows the content of `tasks.module.ts`:

```
1 import { Module } from '@nestjs/common';
2
3 @Module({})
4 export class TasksModule {}
5
```

TERMINAL: The terminal shows the command used to generate the module:

```
PS C:\Aggio\nestjs\primeiro-app> nest g module tasks
CREATE src/tasks/tasks.module.ts (86 bytes)
```

PROBLEMS: The problems panel is empty.

OUTPUT: The output panel is empty.

DEBUG CONSOLE: The debug console is empty.

PORTS: The ports panel is empty.

SCHEMATICS: A table of available schematics is displayed:

name	alias	description
application	application	Generate a new application workspace
class	cl	Generate a new class
configuration	config	Generate a CLI configuration file
controller	co	Generate a controller declaration
decorator	d	Generate a custom decorator
filter	f	Generate a filter declaration
gateway	ga	Generate a gateway declaration
guard	gu	Generate a guard declaration
interceptor	itc	Generate an interceptor declaration
interface	itf	Generate an interface
library	lib	Generate a new library within a monorepo
middleware	mi	Generate a middleware declaration
module	mo	Generate a module declaration
pipe	pi	Generate a pipe declaration
provider	pr	Generate a provider declaration
resolver	r	Generate a GraphQL resolver declaration
resource	res	Generate a new CRUD resource
service	s	Generate a service declaration
sub-app	app	Generate a new application within a monorepo

The screenshot shows the Visual Studio Code editor with the file explorer on the left, the editor window in the center, and the terminal at the bottom. The file explorer shows the project structure for 'PRIMEIRO-APP', including 'src/app' and 'src/tasks'. The editor window displays the 'app.module.ts' file, which imports 'TasksModule' from '../tasks/tasks.module'. The terminal shows the output of the 'nest-cli' command, which lists various CLI commands and their descriptions.

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - app.controller.ts
 - app.module.ts
 - app.service.ts
 - tasks
 - tasks.module.ts
 - app.controller.spec.ts
 - main.ts
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.build.json
 - tsconfig.json

app.module.ts

```

1 import { Module } from '@nestjs/common';
2 import { AppController } from './app.controller';
3 import { AppService } from './app.service';
4 import { TasksModule } from '../tasks/tasks.module';
5
6 @Module({
7   imports: [TasksModule],
8   controllers: [AppController],
9   providers: [AppService],
10 })
11 export class AppModule {}
12
  
```

TERMINAL

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL	PORTS
	application	application	Generate a new application workspace	
	class	cl	Generate a new class	
	configuration	config	Generate a CLI configuration file	
	controller	co	Generate a controller declaration	
	decorator	d	Generate a custom decorator	
	filter	f	Generate a filter declaration	
	gateway	ga	Generate a gateway declaration	
	guard	gu	Generate a guard declaration	
	interceptor	itc	Generate an interceptor declaration	
	interface	itf	Generate an interface	
	library	lib	Generate a new library within a monorepo	

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - app.controller.ts
 - app.module.ts**
 - app.service.ts
 - tasks
 - tasks.module.ts
 - app.controller.spec.ts
 - main.ts
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.build.json
 - tsconfig.json

app.module.ts

```

1  import { Module } from '@nestjs/common';
2  import { AppController } from './app.controller';
3  import { AppService } from './app.service';
4  import { TasksModule } from '../tasks/tasks.module';
5
6  @Module({
7    imports: [TasksModule],
8    controllers: [AppController],
9    providers: [AppService],
10 })
11 export class AppModule {}
12

```

TERMINAL

```

[18:14:47] Starting compilation in watch mode...
[18:14:49] Found 0 errors. Watching for file changes.
[Nest] 10476 - 06/02/2025, 18:14:49 LOG [NestFactory] Starting Nest application...
[Nest] 10476 - 06/02/2025, 18:14:49 LOG [InstanceLoader] TasksModule dependencies initialized +8ms
[Nest] 10476 - 06/02/2025, 18:14:49 LOG [InstanceLoader] AppModule dependencies initialized +0ms
[Nest] 10476 - 06/02/2025, 18:14:49 LOG [RoutesResolver] AppController {/}: +4ms
[Nest] 10476 - 06/02/2025, 18:14:49 LOG [RouterExplorer] Mapped {/, GET} route +3ms
[Nest] 10476 - 06/02/2025, 18:14:49 LOG [RouterExplorer] Mapped {/novarota, GET} route +0ms
[Nest] 10476 - 06/02/2025, 18:14:49 LOG [NestApplication] Nest application successfully started +2ms

```

npm run start:dev

File

Edit

Selection

View

Go

...

←

→

primeiro-app

🔍

🔗

📄

🔌

🔧

👤

⚙️

EXPLORER

PRIMEIRO-APP

dist

node_modules

src

app

app.controller.ts

app.module.ts

app.service.ts

tasks

tasks.module.ts

app.controller.spec.ts

main.ts

test

.gitignore

.prettierrc

eslint.config.mjs

nest-cli.json

package-lock.json

package.json

README.md

tsconfig.build.json

tsconfig.json

OUTLINE

TIMELINE

app.module.ts

main.ts

tasks.module.ts

src > main.ts > ...

1 import { NestFactory } from '@nestjs/core';

2 import { AppModule } from './app/app.module';

3

4 /*

5 - app.module.ts: É o módulo principal do aplicativo

6 - app.controller.ts: Define as rotas e lida com as requisições

7 - app.service.ts: Contém a lógica do negócio, separado do controlador.

8 */

9

10 //Arquivo que inicia o nosso projeto

11 async function bootstrap() {

12 const app = await NestFactory.create(AppModule);

13 await app.listen(process.env.PORT ?? 3000);

14 }

15 bootstrap();

16

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

node

+

🗑️

...

^

✕

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:14:47] Starting compilation in watch mode...

[18:19:38] File change detected. Starting incremental compilation...

Ln 3, Col 1 (203 selected)

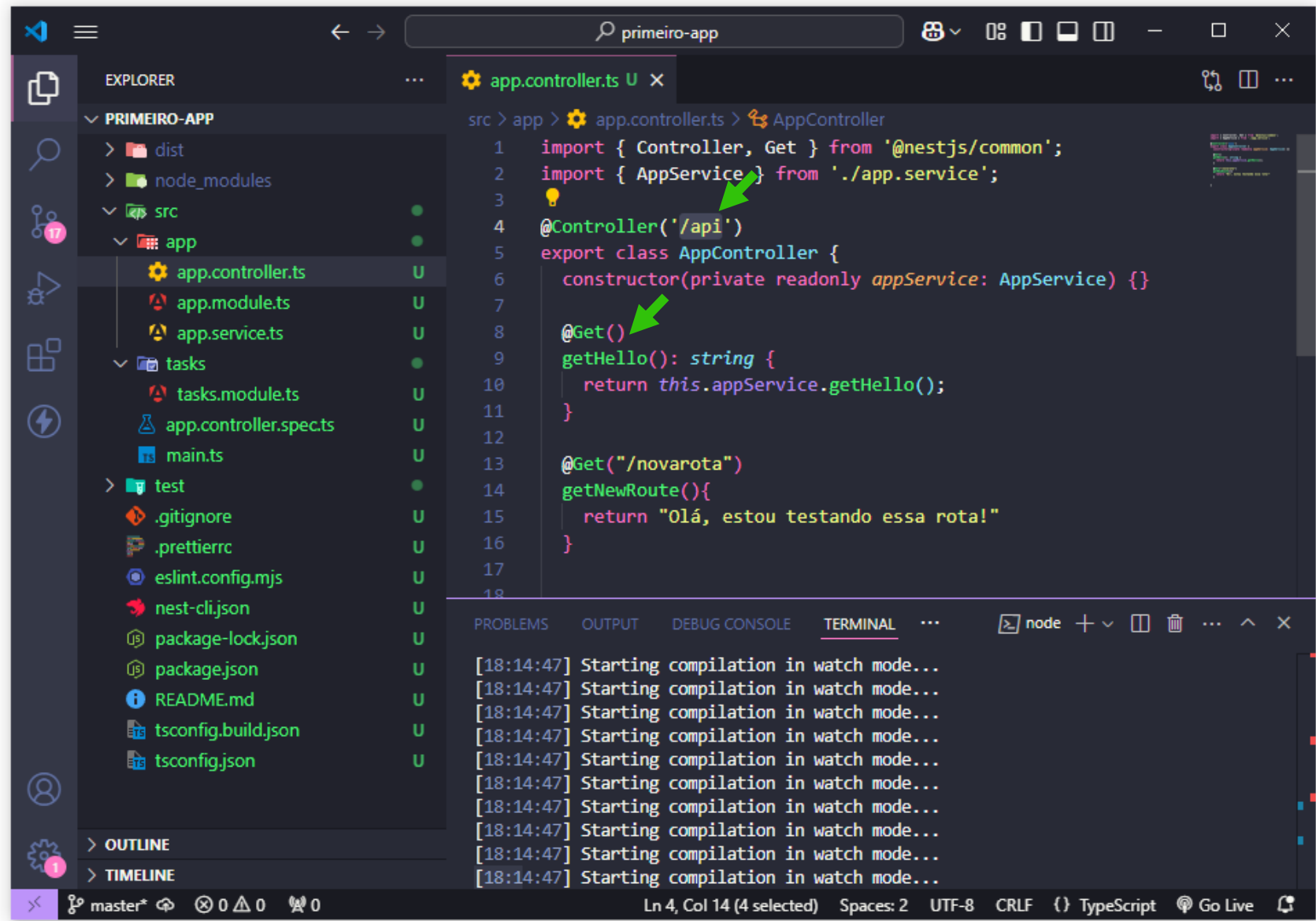
Spaces: 2

UTF-8

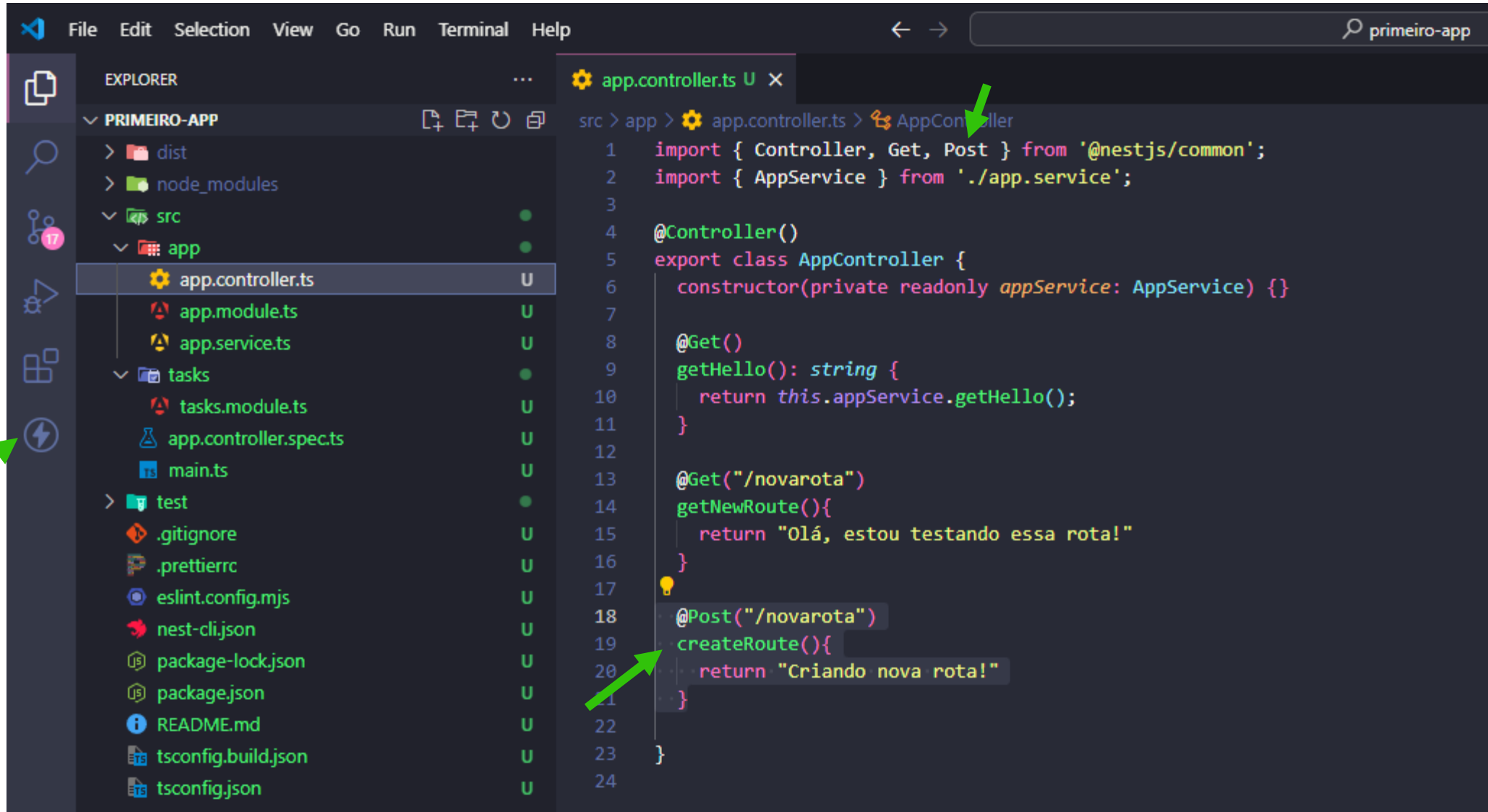
CRLF

{ } TypeScript

Go Live



Lidando com Controller

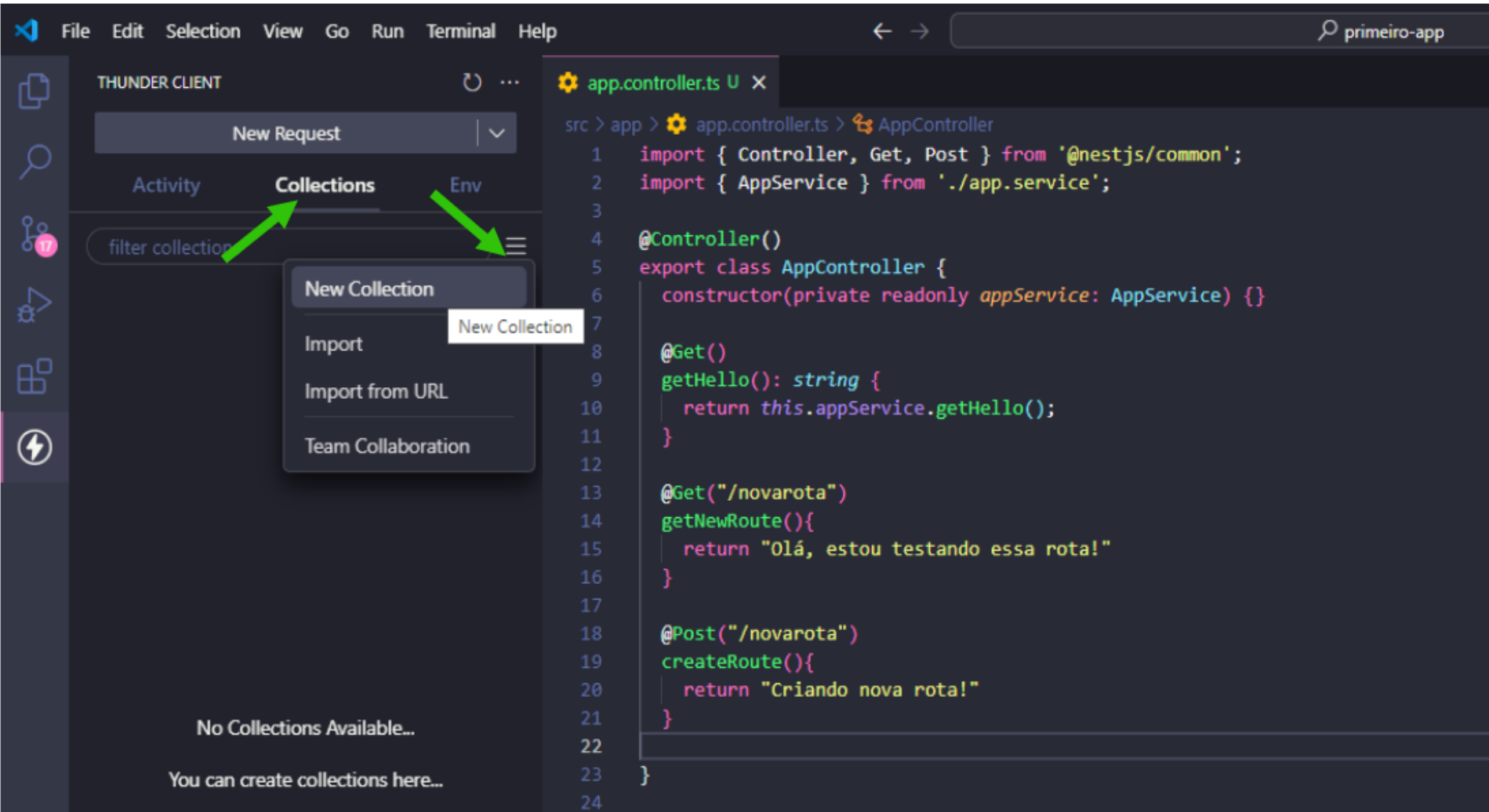


```
File Edit Selection View Go Run Terminal Help
primeiro-app

EXPLORER
PRIMEIRO-APP
  dist
  node_modules
  src
    app
      app.controller.ts
      app.module.ts
      app.service.ts
    tasks
      tasks.module.ts
      app.controller.spec.ts
      main.ts
    test
      .gitignore
      .prettierrc
      eslint.config.mjs
      nest-cli.json
      package-lock.json
      package.json
      README.md
      tsconfig.build.json
      tsconfig.json

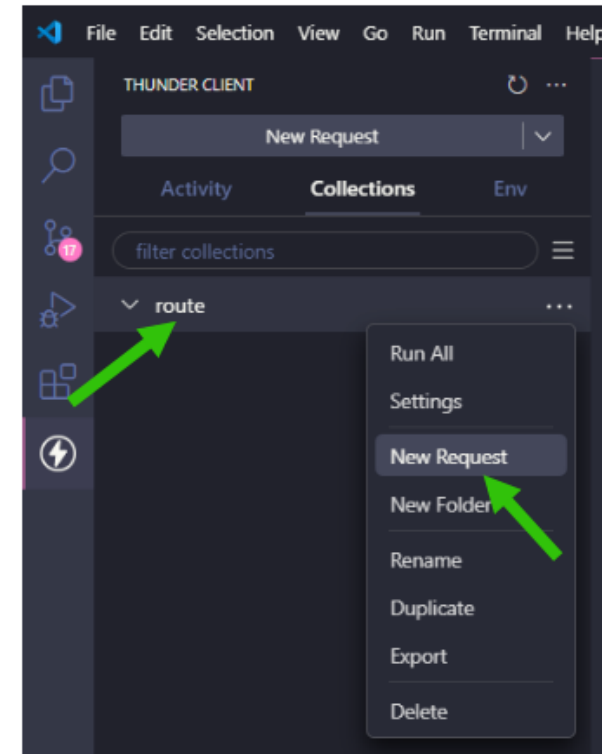
src > app > app.controller.ts > AppController
1 import { Controller, Get, Post } from '@nestjs/common';
2 import { AppService } from './app.service';
3
4 @Controller()
5 export class AppController {
6   constructor(private readonly appService: AppService) {}
7
8   @Get()
9   getHello(): string {
10     return this.appService.getHello();
11   }
12
13   @Get("/novarota")
14   getNewRoute(){
15     return "Olá, estou testando essa rota!"
16   }
17
18   @Post("/novarota")
19   createRoute(){
20     return "Criando nova rota!"
21   }
22
23 }
24
```

Lidando com Controller



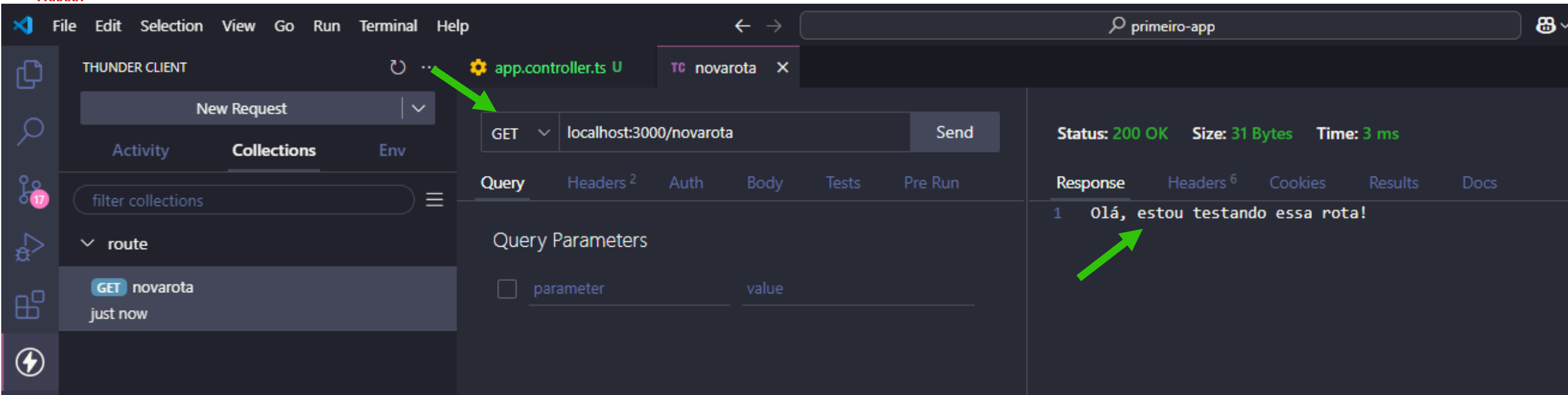
The screenshot shows the Thunder Client interface with the 'Collections' tab selected. A green arrow points to the 'New Collection' button in the 'Collections' tab. Another green arrow points to the 'New Collection' button in the dropdown menu. The code editor displays the following code:

```
src > app > app.controller.ts > AppController
1 import { Controller, Get, Post } from '@nestjs/common';
2 import { AppService } from '../app.service';
3
4 @Controller()
5 export class AppController {
6   constructor(private readonly appService: AppService) {}
7
8   @Get()
9   getHello(): string {
10     return this.appService.getHello();
11   }
12
13   @Get("/novarota")
14   getNewRoute(){
15     return "Olá, estou testando essa rota!"
16   }
17
18   @Post("/novarota")
19   createRoute(){
20     return "Criando nova rota!"
21   }
22 }
23
24
```



The screenshot shows the Thunder Client interface with the 'Collections' tab selected. A green arrow points to the 'route' collection. A context menu is open, showing options: Run All, Settings, New Request, New Folder, Rename, Duplicate, Export, and Delete. A green arrow points to the 'New Folder' option.

Lidando com Controller



THUNDER CLIENT

File Edit Selection View Go Run Terminal Help

primeiro-app

app.controller.ts U TC novarota X

New Request

Activity Collections Env

filter collections

route

GET novarota just now

GET localhost:3000/novarota Send

Query Headers² Auth Body Tests Pre Run

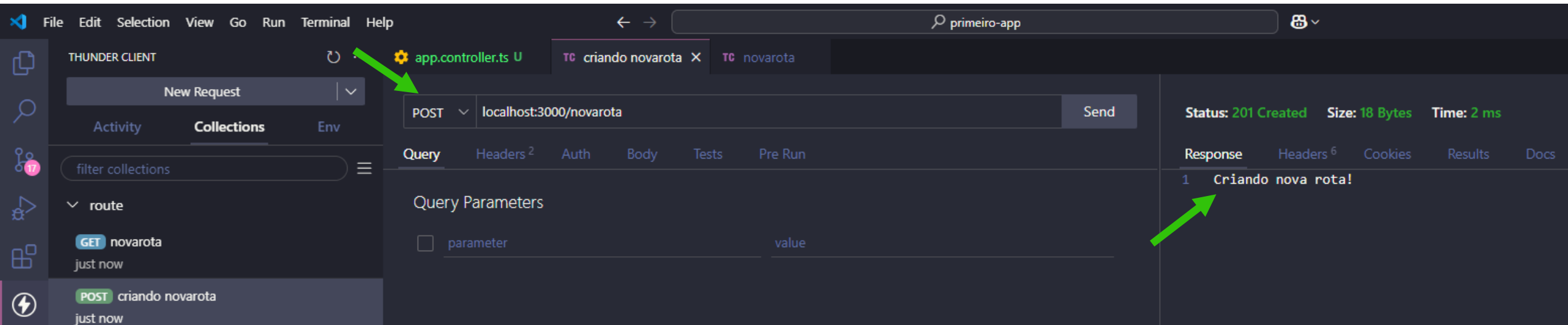
Query Parameters

parameter value

Status: 200 OK Size: 31 Bytes Time: 3 ms

Response Headers⁶ Cookies Results Docs

1 Olá, estou testando essa rota!



THUNDER CLIENT

File Edit Selection View Go Run Terminal Help

primeiro-app

app.controller.ts U TC criando novarota X TC novarota

New Request

Activity Collections Env

filter collections

route

GET novarota just now

POST criando novarota just now

POST localhost:3000/novarota Send

Query Headers² Auth Body Tests Pre Run

Query Parameters

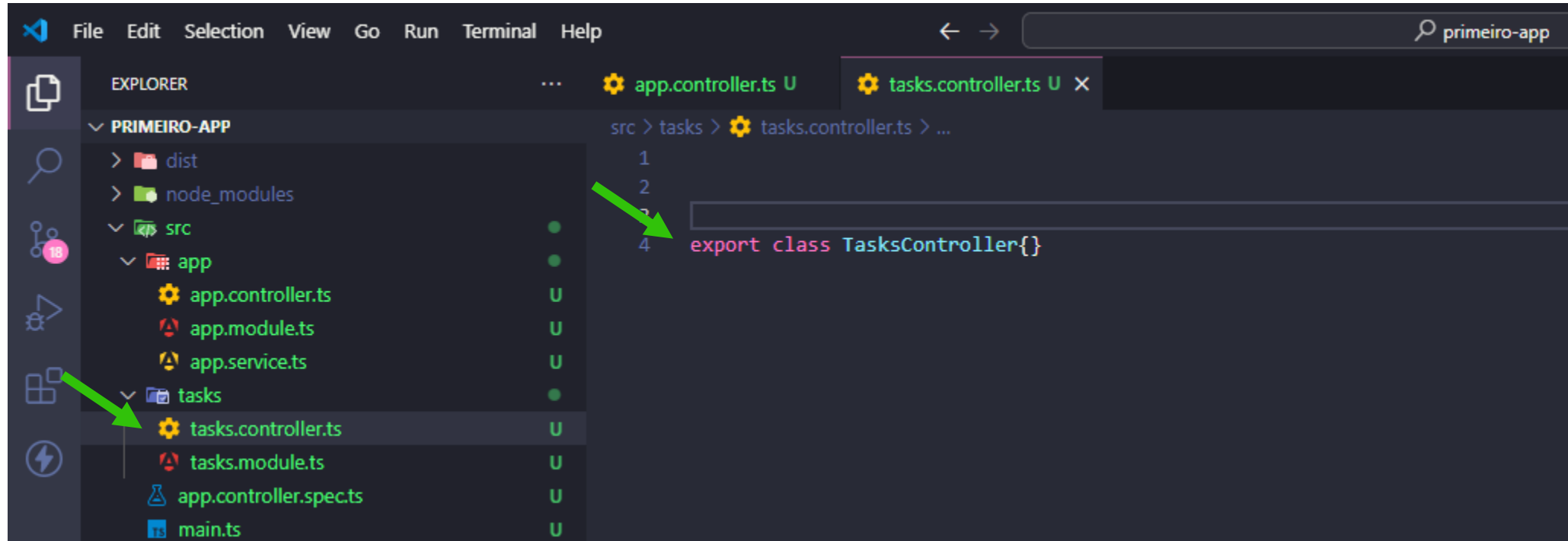
parameter value

Status: 201 Created Size: 18 Bytes Time: 2 ms

Response Headers⁶ Cookies Results Docs

1 Criando nova rota!

Gerando Controller



Gerando Controller

VS Code Explorer: PRIMEIRO-APP

- dist
- node_modules
- src
 - app
 - app.controller.ts
 - app.module.ts
 - app.service.ts
 - tasks
 - tasks.controller.ts (1, U)
 - tasks.module.ts
- app.controller.spec.ts
- main.ts
- test
- .gitignore
- .prettierrc
- eslint.config.mjs
- nest-cli.json
- package-lock.json
- package.json
- README.md
- tsconfig.build.json
- tsconfig.json

Editor: src > tasks > tasks.controller.ts > TasksController

```

1
2
3 @Contr
4 export

```

Autocomplete dropdown for `@Controller` from `@nestjs/common`:

- `CONTROLLER_WATERMARK` (@nestjs/common/constants)
- `ContextCreator` (@nestjs/core/helpers/context-creator)
- `constrainedMemory` (process)
- `constrainedMemory` (node:process)
- `ContextIdFactory` (@nestjs/core)
- `countReset` (console)
- `countReset` (node:console)
- `AppController` (src/app/app.controller)
- `ConstantSourceNode`
- `TasksController`
- `AbortController`

Right Sidebar: Add import from "@nestjs/common"

function Controller(): ClassDecorator ...

Decorator that marks a class as a Nest controller that can receive inbound requests and produce responses.

An HTTP Controller responds to inbound HTTP Requests and produces HTTP Responses. It defines a class that provides the context for one or more related route handlers that correspond to HTTP request methods and associated routes for example GET /api/profile , POST /users/resume .

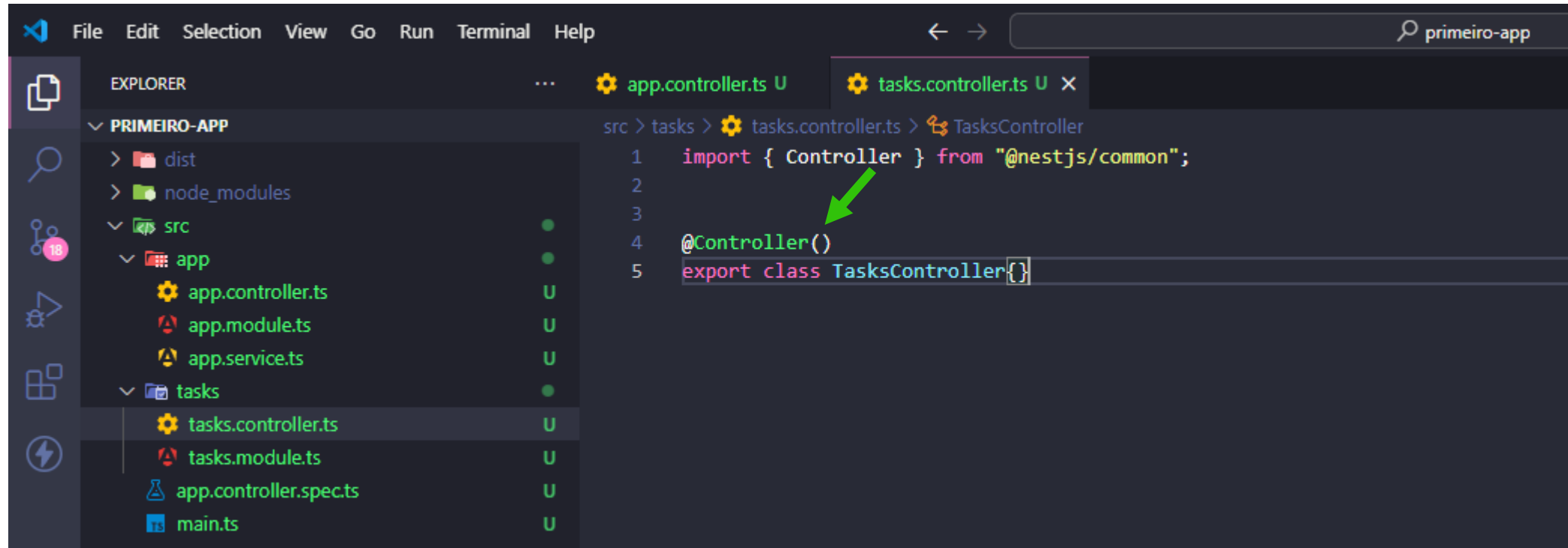
A Microservice Controller responds to requests as well as events, running over a variety of transports (read more here). It defines a class that provides a context for one or more message or event handlers.

@see — Controllers

@see — Microservices

@publicApi

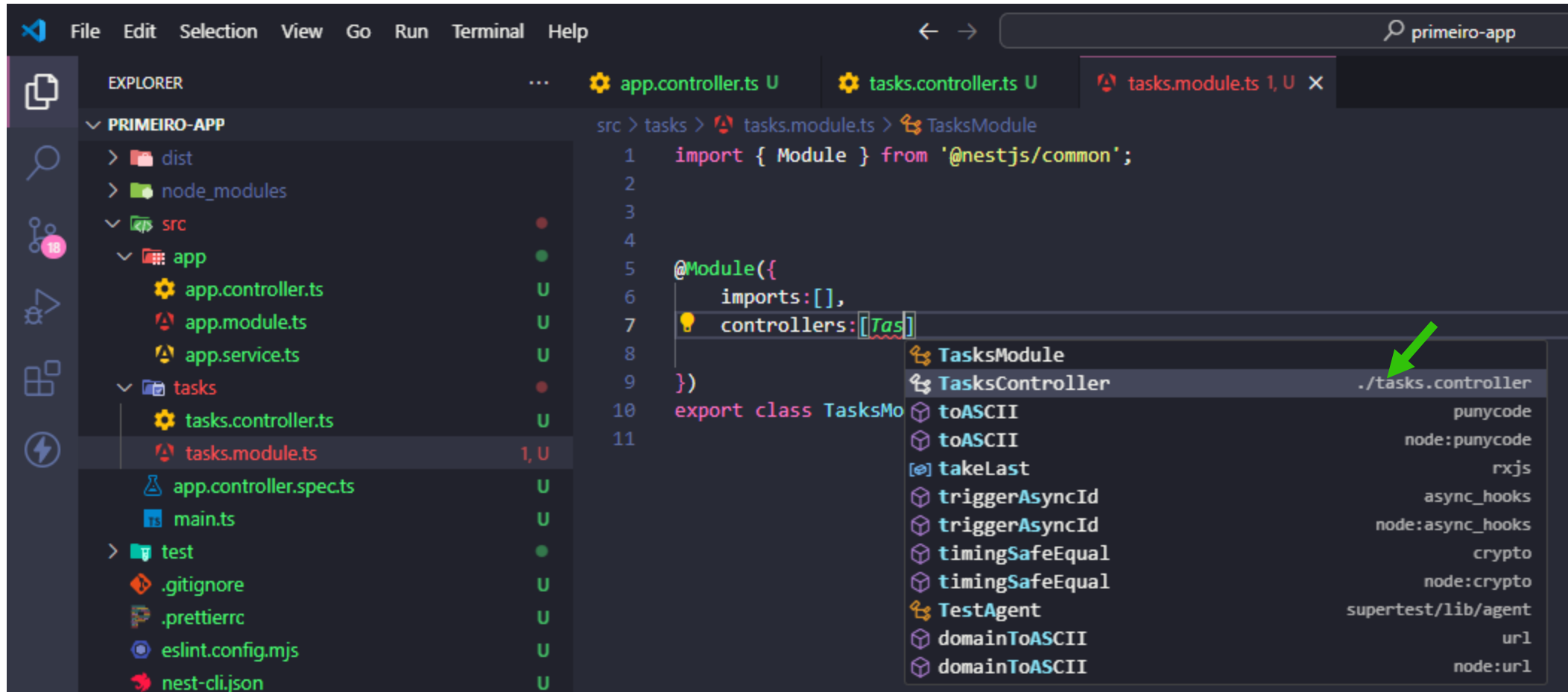
Gerando Controller



The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left and the Editor window on the right. The Explorer sidebar shows the project structure for 'PRIMEIRO-APP', including folders like 'dist', 'node_modules', 'src', and 'tasks'. The 'tasks' folder is expanded, showing 'tasks.controller.ts' selected. The Editor window displays the code for 'tasks.controller.ts', which includes an import statement for 'Controller' from '@nestjs/common', a decorator '@Controller()', and the start of a class definition 'export class TasksController{}'. A green arrow points to the '@Controller()' decorator.

```
1 import { Controller } from "@nestjs/common";
2
3
4 @Controller()
5 export class TasksController{}
```

Gerando Controller

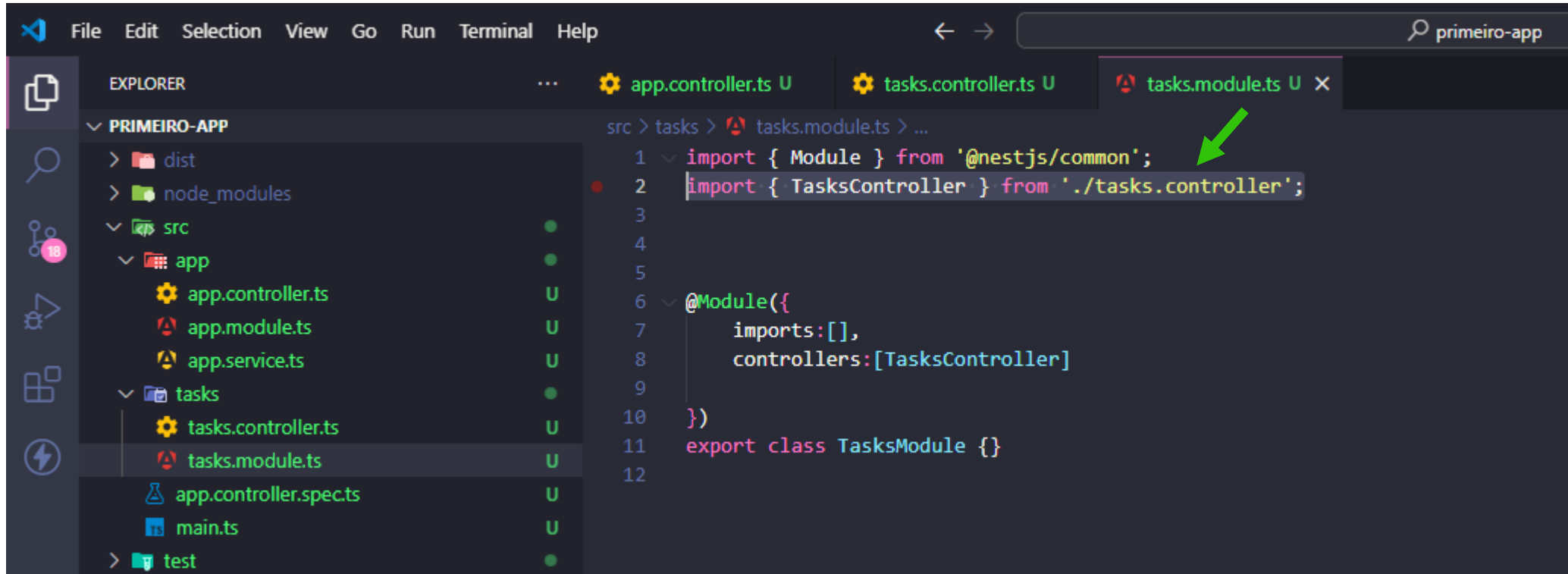


The screenshot shows the Visual Studio Code interface with a project named "PRIMEIRO-APP". The Explorer sidebar on the left displays the file structure, including a "tasks" directory. The "tasks.module.ts" file is selected, and the "tasks.controller.ts" file is highlighted in the Explorer. The main editor shows the code for "tasks.module.ts", which is a NestJS module. The code includes an import for "Module" from "@nestjs/common", a decorator "@Module" with "imports: []" and "controllers: [TasksController]", and an export for the "TasksModule" class. A green arrow points to the "TasksController" entry in the "controllers" array, indicating the controller to be generated. A dropdown menu is open, showing a list of available controllers, with "TasksController" selected. The dropdown also lists other controllers like "toASCII", "takeLast", "triggerAsyncId", "timingSafeEqual", "TestAgent", "domainToASCII", and "domainToASCII", along with their respective modules (e.g., "punycode", "node:punycode", "rxjs", "async_hooks", "node:async_hooks", "crypto", "node:crypto", "supertest/lib/agent", "url", "node:url").

```
1 import { Module } from '@nestjs/common';
2
3
4
5 @Module({
6   imports: [],
7   controllers: [TasksController]
8 })
9
10 export class TasksModule {}
11
```

TasksModule
TasksController ./tasks.controller
toASCII punycode
toASCII node:punycode
takeLast rxjs
triggerAsyncId async_hooks
triggerAsyncId node:async_hooks
timingSafeEqual crypto
timingSafeEqual node:crypto
TestAgent supertest/lib/agent
domainToASCII url
domainToASCII node:url

Gerando Controller



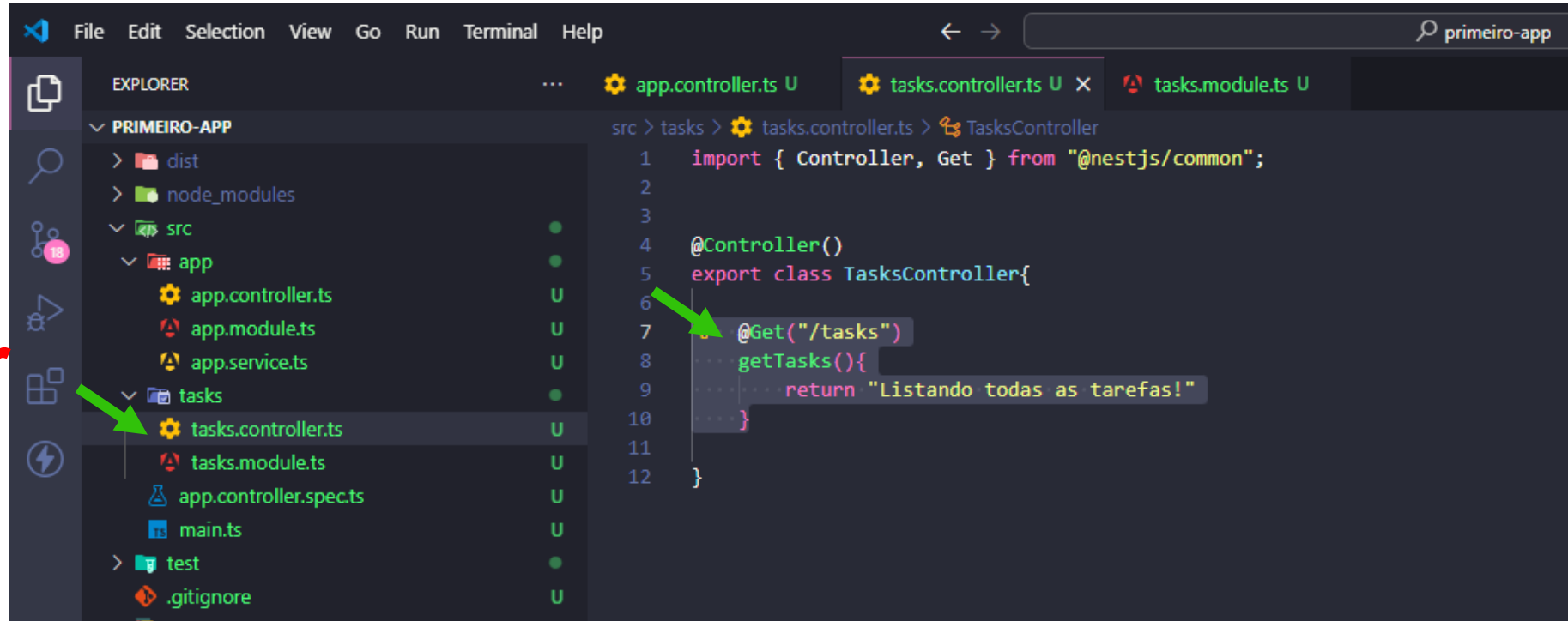
The screenshot shows the Visual Studio Code editor with the Explorer sidebar on the left and the Editor view on the right. The Explorer sidebar displays the file structure of a project named "PRIMEIRO-APP". The file tree includes a "src" directory with an "app" subdirectory and a "tasks" subdirectory. The "tasks" subdirectory contains "tasks.controller.ts" and "tasks.module.ts". The Editor view shows the "tasks.module.ts" file, which is currently selected. The code in the editor is as follows:

```
1 import { Module } from '@nestjs/common';
2 import { TasksController } from './tasks.controller';
3
4
5
6 @Module({
7   imports:[],
8   controllers:[TasksController]
9 })
10
11 export class TasksModule {}
12
```

A green arrow points to the import statement on line 2: `import { TasksController } from './tasks.controller';`. The Explorer sidebar also shows the status of the files: "app.controller.ts" is updated (U), "app.module.ts" is updated (U), "app.service.ts" is updated (U), "tasks.controller.ts" is updated (U), and "tasks.module.ts" is updated (U). The Explorer sidebar also shows the status of the files: "app.controller.spec.ts" is updated (U), "main.ts" is updated (U), and "test" is updated (U).

Gerando Controller

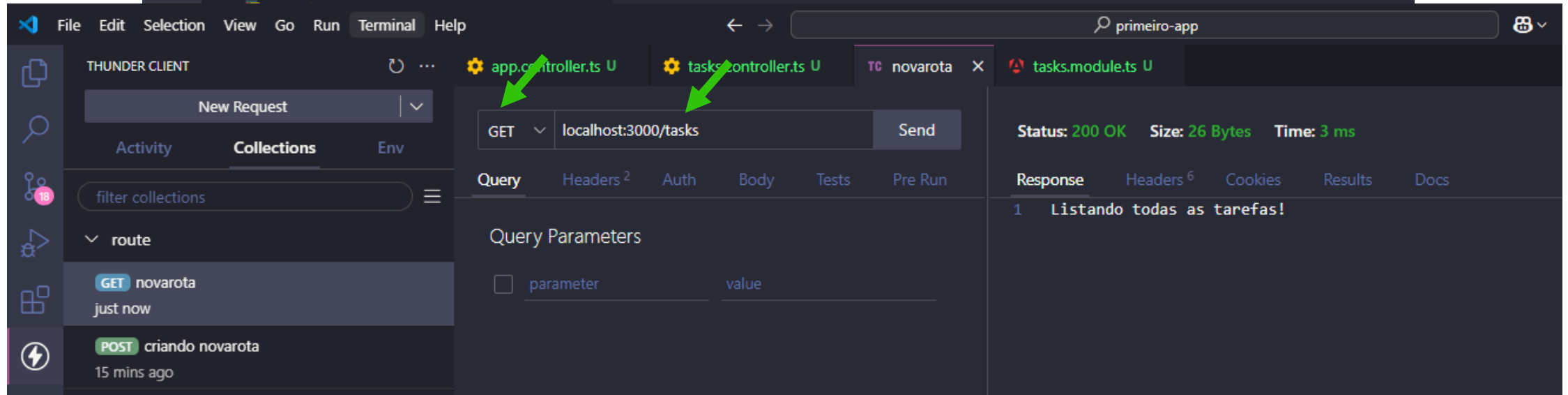
Deletar



VS Code interface showing the file explorer on the left and the editor on the right. The file explorer shows the project structure for 'PRIMEIRO-APP'. The editor shows the code for 'tasks.controller.ts'.

```

1  import { Controller, Get } from "@nestjs/common";
2
3
4  @Controller()
5  export class TasksController{
6
7    @Get("/tasks")
8    getTasks(){
9      return "Listando todas as tarefas!";
10   }
11
12 }
```



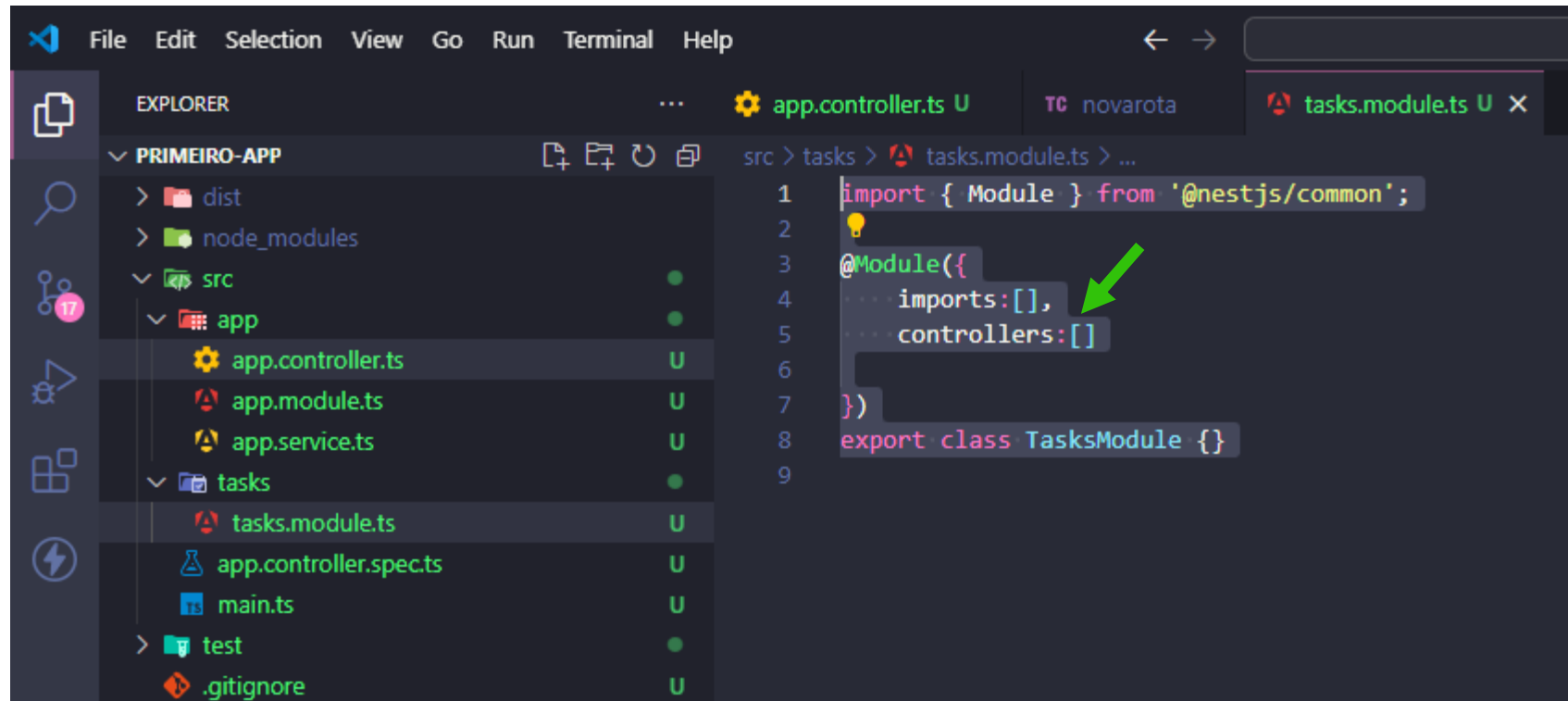
Thunder Client interface showing a GET request to localhost:3000/tasks. The response is 'Listando todas as tarefas!'.

Request: GET localhost:3000/tasks

Status: 200 OK Size: 26 Bytes Time: 3 ms

Response: 1 Listando todas as tarefas!

Gerando Controller com CLI



```
File Edit Selection View Go Run Terminal Help
```

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - app.controller.ts U
 - app.module.ts U
 - app.service.ts U
 - tasks
 - tasks.module.ts U
 - app.controller.spec.ts U
 - main.ts U
 - test
 - .gitignore U

src > tasks > tasks.module.ts > ...

```
1 import { Module } from '@nestjs/common';
2
3 @Module({
4   ...imports:[],
5   ...controllers:[]
6 })
7
8 export class TasksModule {}
9
```

Gerando Controller com CLI

Visual Studio Code interface showing the Explorer, Editor, and Terminal panels.

EXPLORER: The file explorer on the left shows the project structure for 'PRIMEIRO-APP'. The 'tasks' folder is expanded, showing 'tasks.module.ts' selected.

EDITOR: The editor shows the content of 'tasks.module.ts':

```
1 import { Module } from '@nestjs/common';
2
3 @Module({
4   imports:[],
5   controllers:[]
6 })
7
8 export class TasksModule {}
9
```

TERMINAL: The terminal shows the command 'nest --help' and its output:

```
PS C:\Aggio\nestjs\primeiro-app> nest --help
Usage: nest <command> [options]

Options:
  -v, --version           Output the current version.
  -h, --help              Output usage information.

Commands:
  new|n [options] [name]  Generate Nest application.
  build [options] [apps...] Build Nest application.
  start [options] [app]   Run Nest application.
  info|i                 Display Nest project details.
  add [options] <library> Adds support for an external library to your project.
  generate|g [options] <schematic> [name] [path] Generate a Nest element.
                        Schematics available on @nestjs/schematics collection:
```

A table is displayed in the terminal, listing available schematics:

name	alias	description
application	application	Generate a new application workspace
class	cl	Generate a new class
configuration	config	Generate a CLI configuration file
controller	co	Generate a controller declaration
decorator	d	Generate a custom decorator
filter	f	Generate a filter declaration
gateway	ga	Generate a gateway declaration
guard	gu	Generate a guard declaration
interceptor	itc	Generate an interceptor declaration
interface	itf	Generate an interface
library	lib	Generate a new library within a monorepo
middleware	mi	Generate a middleware declaration
module	mo	Generate a module declaration
pipe	pi	Generate a pipe declaration
provider	pr	Generate a provider declaration
resolver	r	Generate a GraphQL resolver declaration
resource	res	Generate a new CRUD resource
service	s	Generate a service declaration
sub-app	app	Generate a new application within a monorepo

At the bottom of the terminal, the command 'nest g controller tasks' is entered.

Gerando Controller com CLI

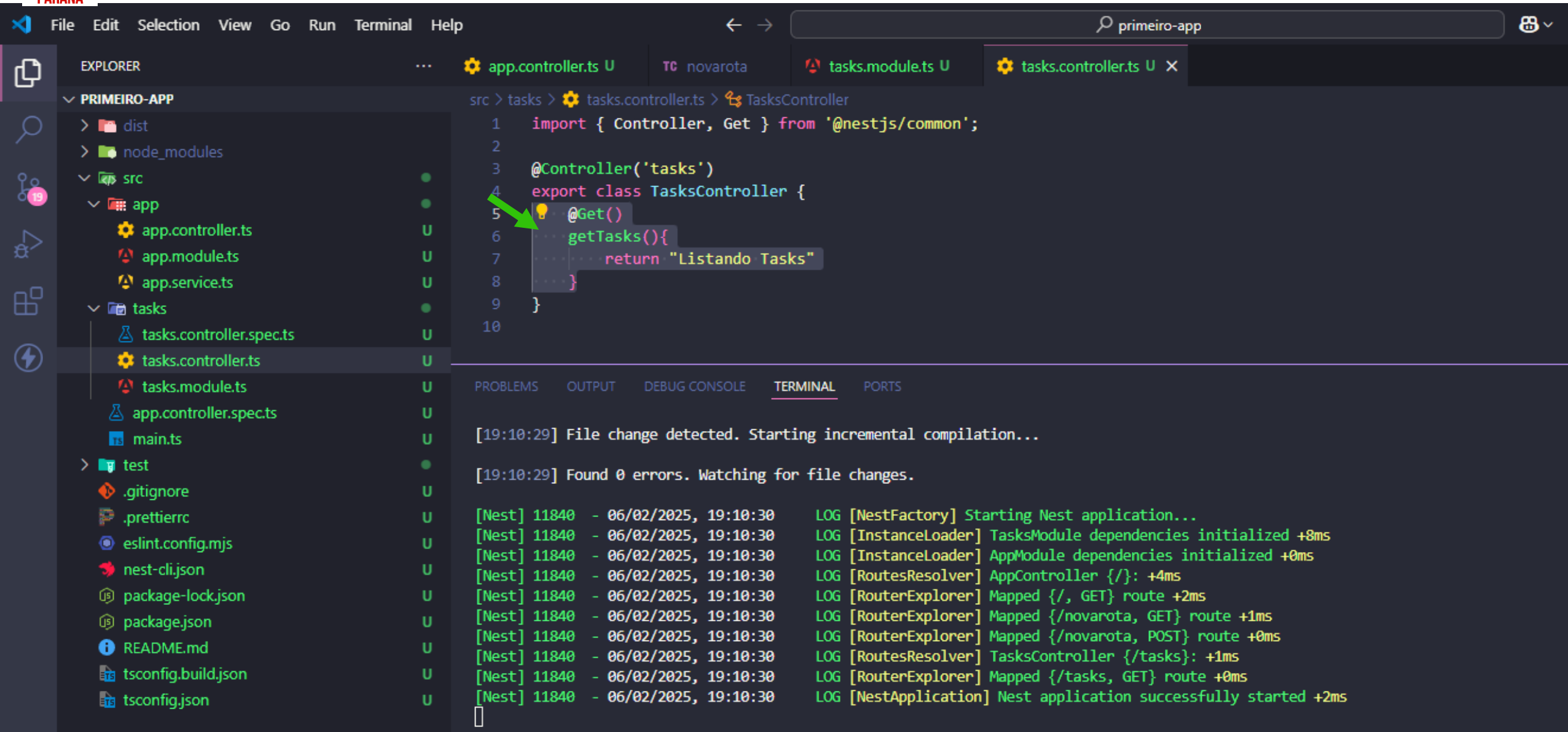


The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** Displays the project structure. The 'tasks' folder is expanded, showing files like 'tasks.controller.spec.ts', 'tasks.controller.ts', and 'tasks.module.ts'. A green arrow points to the 'tasks' folder.
- SOURCE CONTROL:** Shows the changes made by the CLI command. It lists the following changes:
 - CREATE src/tasks/tasks.controller.ts (103 bytes)
 - CREATE src/tasks/tasks.controller.spec.ts (503 bytes)
 - UPDATE src/tasks/tasks.module.ts (196 bytes)
- TERMINAL:** Shows the command 'nest g controller tasks' and its output:


```
PS C:\Aggio\nestjs\primeiro-app> nest g controller tasks
CREATE src/tasks/tasks.controller.ts (103 bytes)
CREATE src/tasks/tasks.controller.spec.ts (503 bytes)
UPDATE src/tasks/tasks.module.ts (196 bytes)
PS C:\Aggio\nestjs\primeiro-app>
```

Gerando Controller com CLI



The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** Displays the file structure of the 'primeiro-app' project. The 'tasks' directory is expanded, showing 'tasks.controller.spec.ts' and 'tasks.controller.ts' (selected).
- EDITOR:** Shows the content of 'tasks.controller.ts'. The code defines a 'TasksController' class with a 'getTasks()' method that returns 'Listando Tasks'. A green arrow points to the '@Get()' decorator.
- TERMINAL:** Shows the output of the application startup. It includes messages about file changes, incremental compilation, and the successful initialization of the Nest application with various routes mapped.

```

src > tasks > tasks.controller.ts > TasksController
1  import { Controller, Get } from '@nestjs/common';
2
3  @Controller('tasks')
4  export class TasksController {
5      @Get()
6      getTasks(){
7          return "Listando Tasks"
8      }
9  }
10
  
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

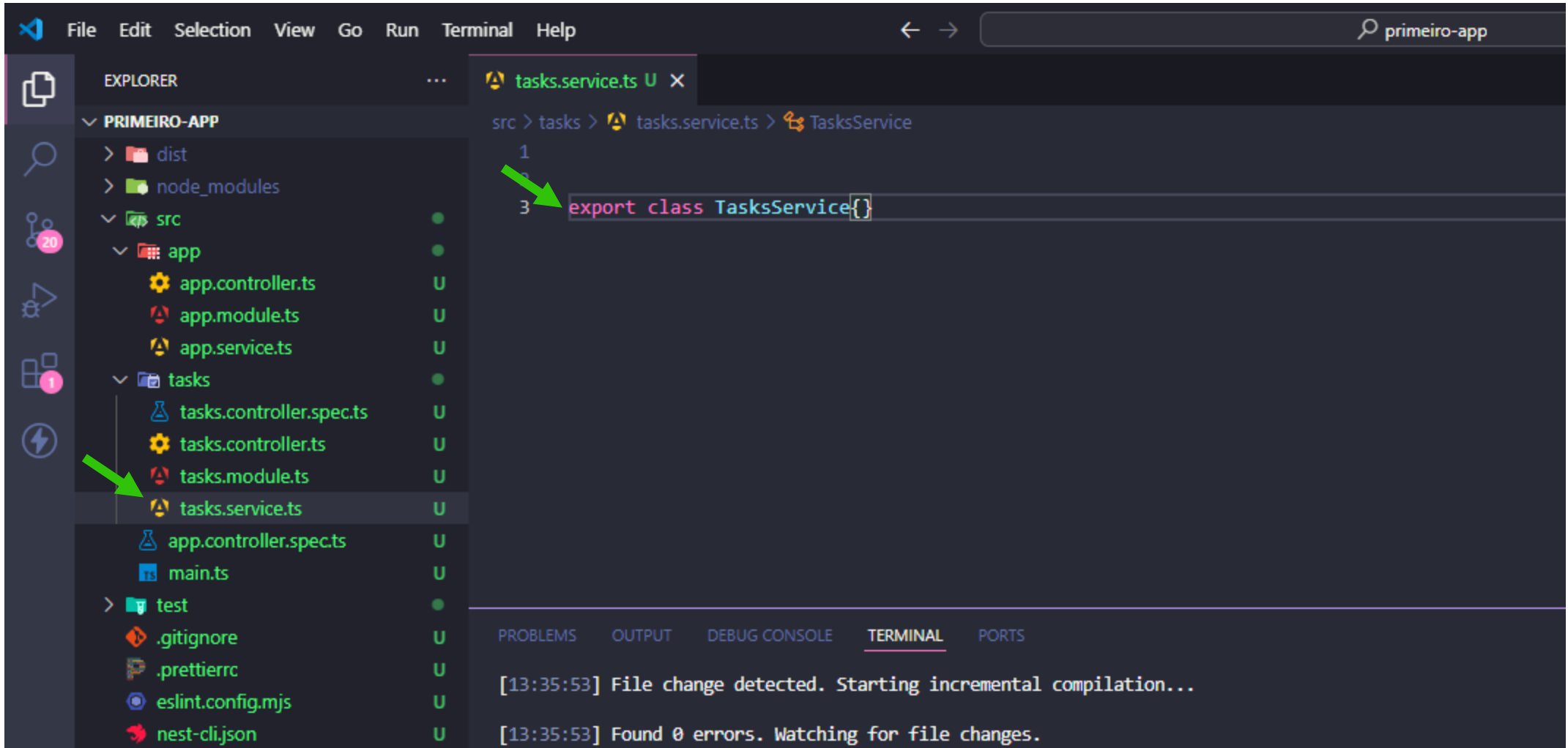
```

[19:10:29] File change detected. Starting incremental compilation...

[19:10:29] Found 0 errors. Watching for file changes.

[Nest] 11840 - 06/02/2025, 19:10:30 LOG [NestFactory] Starting Nest application...
[Nest] 11840 - 06/02/2025, 19:10:30 LOG [InstanceLoader] TasksModule dependencies initialized +8ms
[Nest] 11840 - 06/02/2025, 19:10:30 LOG [InstanceLoader] AppModule dependencies initialized +0ms
[Nest] 11840 - 06/02/2025, 19:10:30 LOG [RoutesResolver] AppController {/}: +4ms
[Nest] 11840 - 06/02/2025, 19:10:30 LOG [RouterExplorer] Mapped {/, GET} route +2ms
[Nest] 11840 - 06/02/2025, 19:10:30 LOG [RouterExplorer] Mapped {/novarota, GET} route +1ms
[Nest] 11840 - 06/02/2025, 19:10:30 LOG [RouterExplorer] Mapped {/novarota, POST} route +0ms
[Nest] 11840 - 06/02/2025, 19:10:30 LOG [RoutesResolver] TasksController {/tasks}: +1ms
[Nest] 11840 - 06/02/2025, 19:10:30 LOG [RouterExplorer] Mapped {/tasks, GET} route +0ms
[Nest] 11840 - 06/02/2025, 19:10:30 LOG [NestApplication] Nest application successfully started +2ms
  
```

Gerando Service / Provider



The screenshot shows the Visual Studio Code interface with the Explorer, Editor, and Terminal views. The Explorer view on the left shows the project structure for 'PRIMEIRO-APP'. The Editor view on the right shows the file 'tasks.service.ts' with the code 'export class TaskService {}'. The Terminal view at the bottom shows the output of the compilation process.

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - app.controller.ts
 - app.module.ts
 - app.service.ts
 - tasks
 - tasks.controller.spec.ts
 - tasks.controller.ts
 - tasks.module.ts
 - tasks.service.ts
 - app.controller.spec.ts
 - main.ts
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json

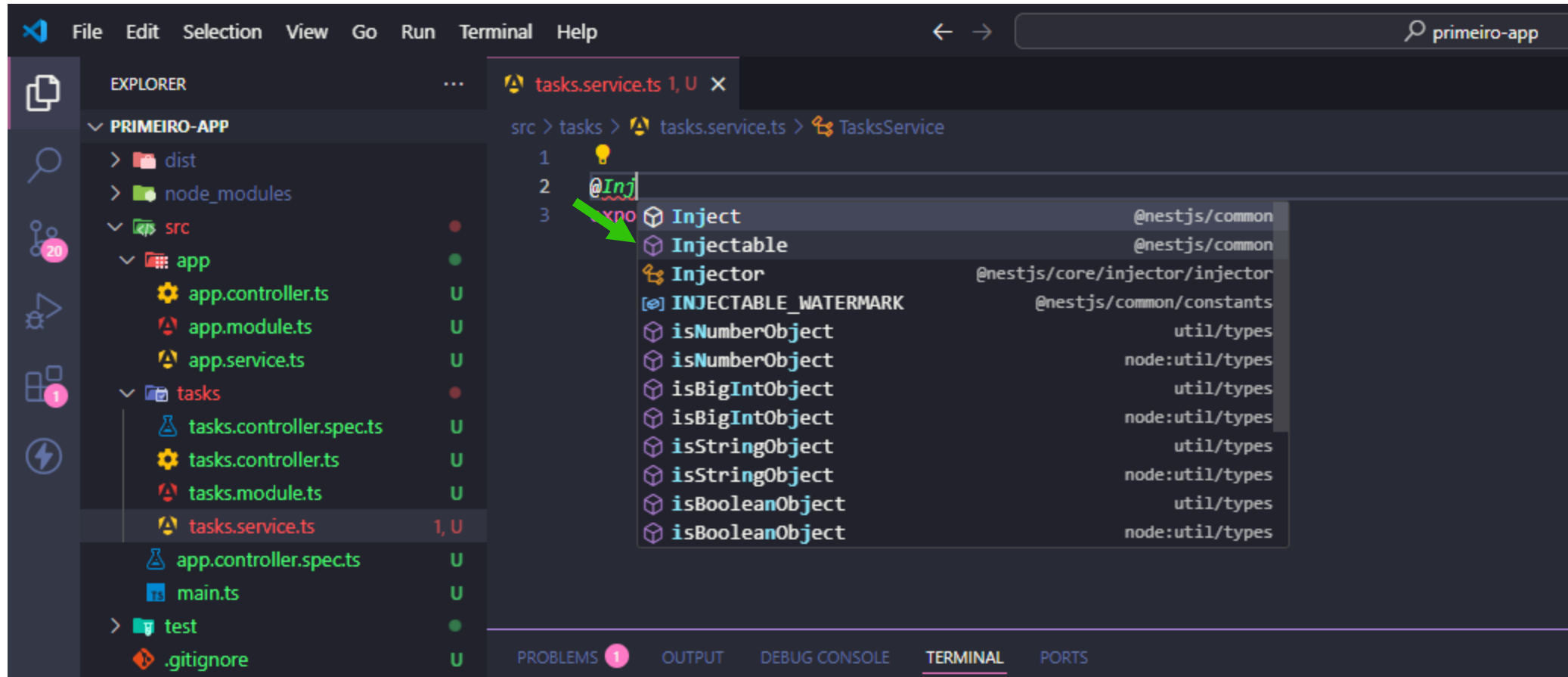
tasks.service.ts

```
1  
2  
3 export class TaskService {}
```

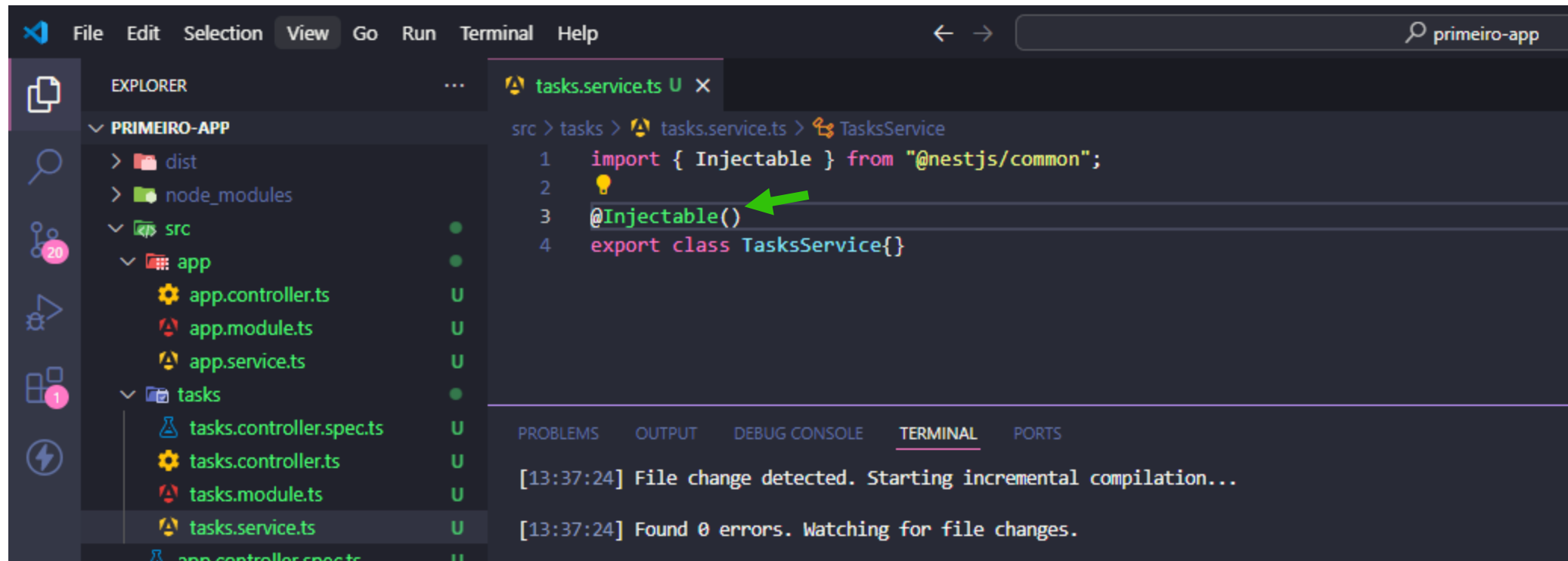
TERMINAL

```
[13:35:53] File change detected. Starting incremental compilation...  
[13:35:53] Found 0 errors. Watching for file changes.
```

Gerando Service / Provider



Gerando Service / Provider



The screenshot shows the Visual Studio Code interface with the Explorer, Editor, and Terminal panels. The Explorer panel on the left shows the project structure for 'primeiro-app', with the 'tasks' folder selected. The Editor panel in the center displays the 'tasks.service.ts' file, which contains the following code:

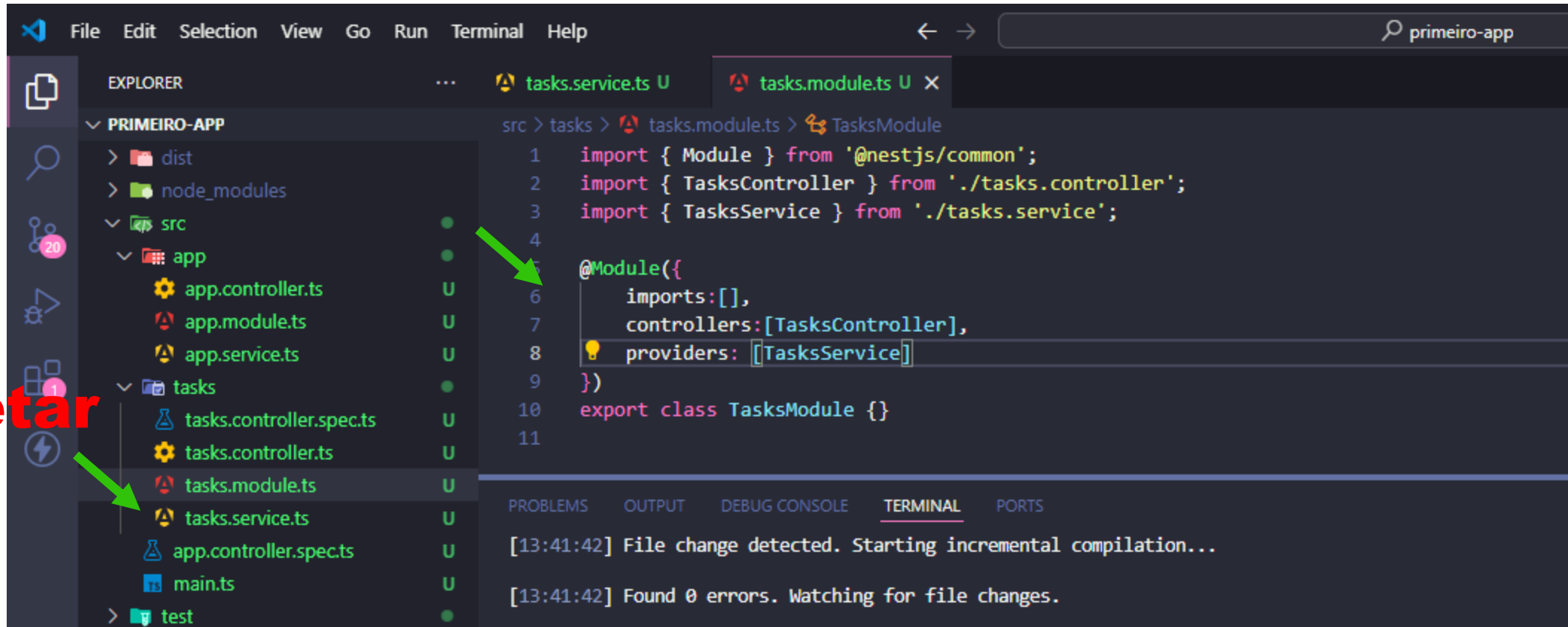
```
1 import { Injectable } from "@nestjs/common";
2
3 @Injectable()
4 export class TaskService{}
```

A green arrow points to the `@Injectable()` decorator on line 3. The Terminal panel at the bottom shows the output of the compilation process:

```
[13:37:24] File change detected. Starting incremental compilation...
[13:37:24] Found 0 errors. Watching for file changes.
```

Gerando Service / Provider

deletar



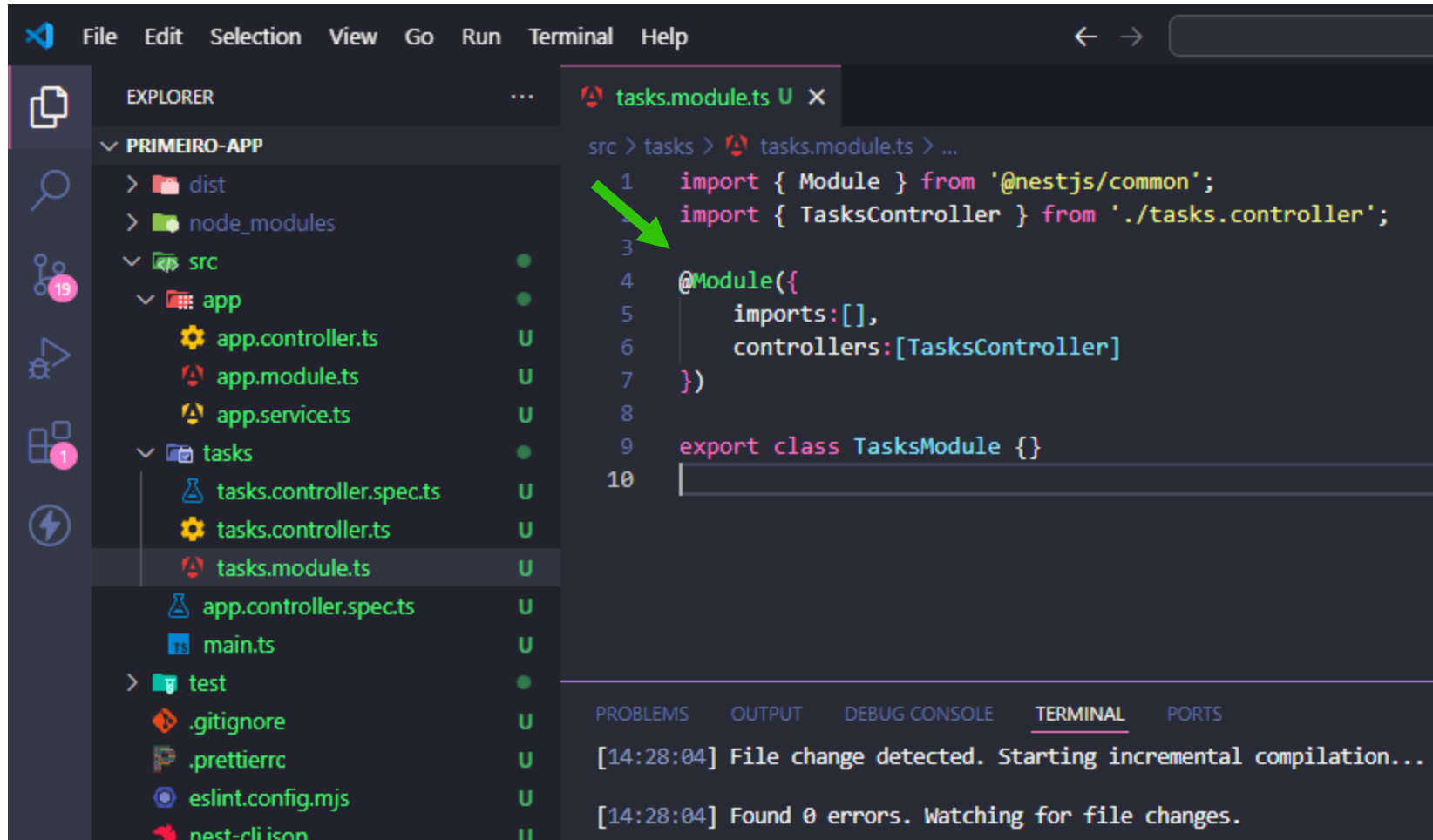
The screenshot shows the Visual Studio Code interface with the Explorer view on the left and the Editor view on the right. The Explorer view shows the file structure of the 'PRIMEIRO-APP' project. The 'tasks' folder is expanded, showing 'tasks.module.ts' and 'tasks.service.ts'. A red arrow points to 'tasks.service.ts' with the word 'deletar' (delete) next to it. The Editor view shows the code in 'tasks.module.ts', which imports 'TasksService' from './tasks.service' and registers it as a provider in the 'TasksModule'.

```
1 import { Module } from '@nestjs/common';
2 import { TasksController } from './tasks.controller';
3 import { TasksService } from './tasks.service';
4
5 @Module({
6   imports: [],
7   controllers: [TasksController],
8   providers: [TasksService]
9 })
10 export class TasksModule {}
11
```

The terminal at the bottom shows the output of the compilation process:

```
[13:41:42] File change detected. Starting incremental compilation...
[13:41:42] Found 0 errors. Watching for file changes.
```

Gerando Service / Provider



```
File Edit Selection View Go Run Terminal Help
```

EXPLORER

PRIMEIRO-APP

- dist
- node_modules
- src
 - app
 - app.controller.ts
 - app.module.ts
 - app.service.ts
 - tasks
 - tasks.controller.spec.ts
 - tasks.controller.ts
 - tasks.module.ts
- test
- .gitignore
- .prettierrc
- eslint.config.mjs
- nest-cli.json

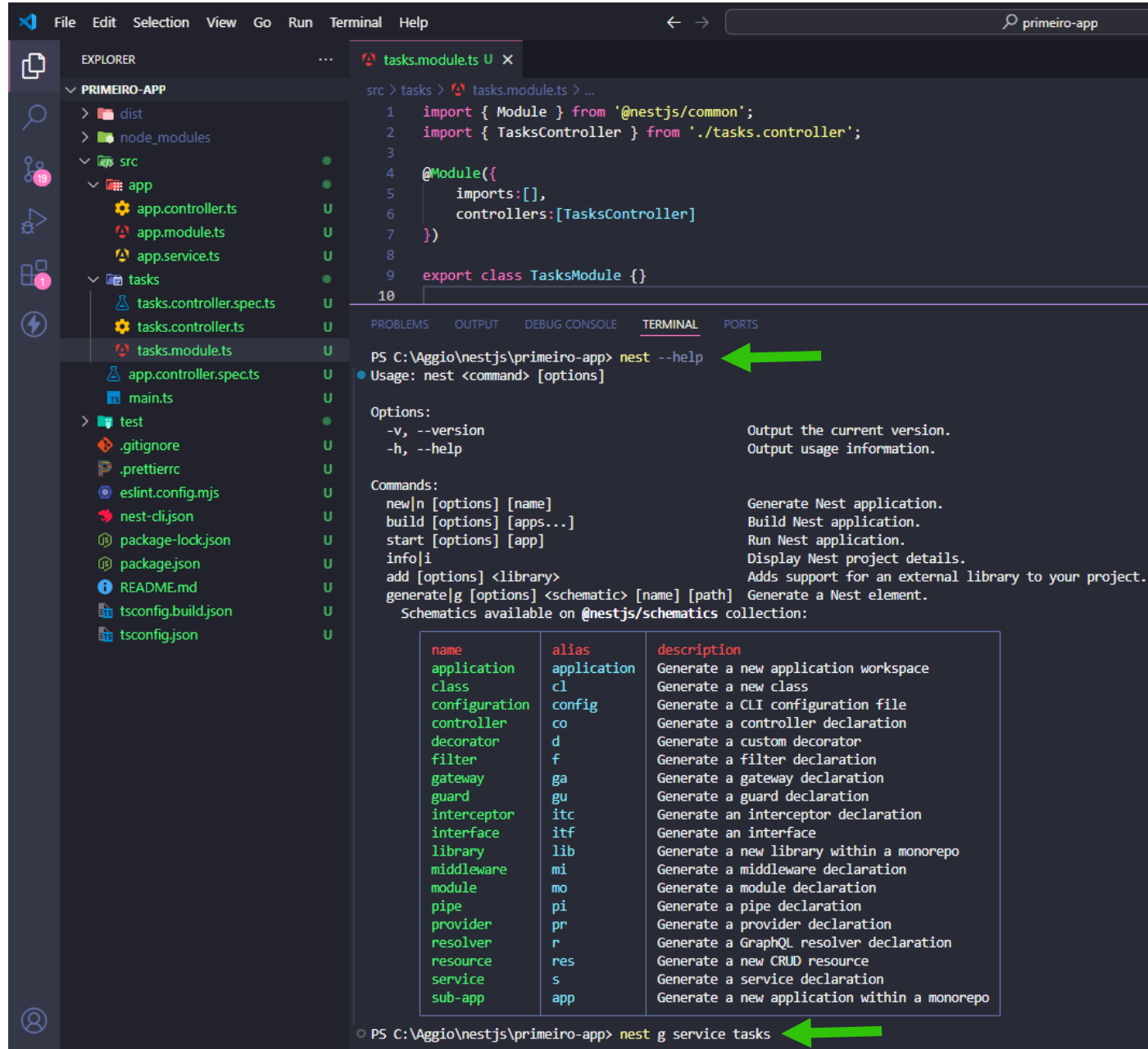
tasks.module.ts

```
src > tasks > tasks.module.ts > ...
1 import { Module } from '@nestjs/common';
2 import { TasksController } from './tasks.controller';
3
4 @Module({
5   imports:[],
6   controllers:[TasksController]
7 })
8
9 export class TasksModule {}
10
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[14:28:04] File change detected. Starting incremental compilation...
[14:28:04] Found 0 errors. Watching for file changes.
```

Gerando Service / Provider



tasks.module.ts

```
1 import { Module } from '@nestjs/common';
2 import { TasksController } from './tasks.controller';
3
4 @Module({
5   imports:[],
6   controllers:[TasksController]
7 })
8
9 export class TasksModule {}
10
```

PS C:\Vaggio\nestjs\primeiro-app> nest --help

Usage: nest <command> [options]

Options:

- v, --version Output the current version.
- h, --help Output usage information.

Commands:

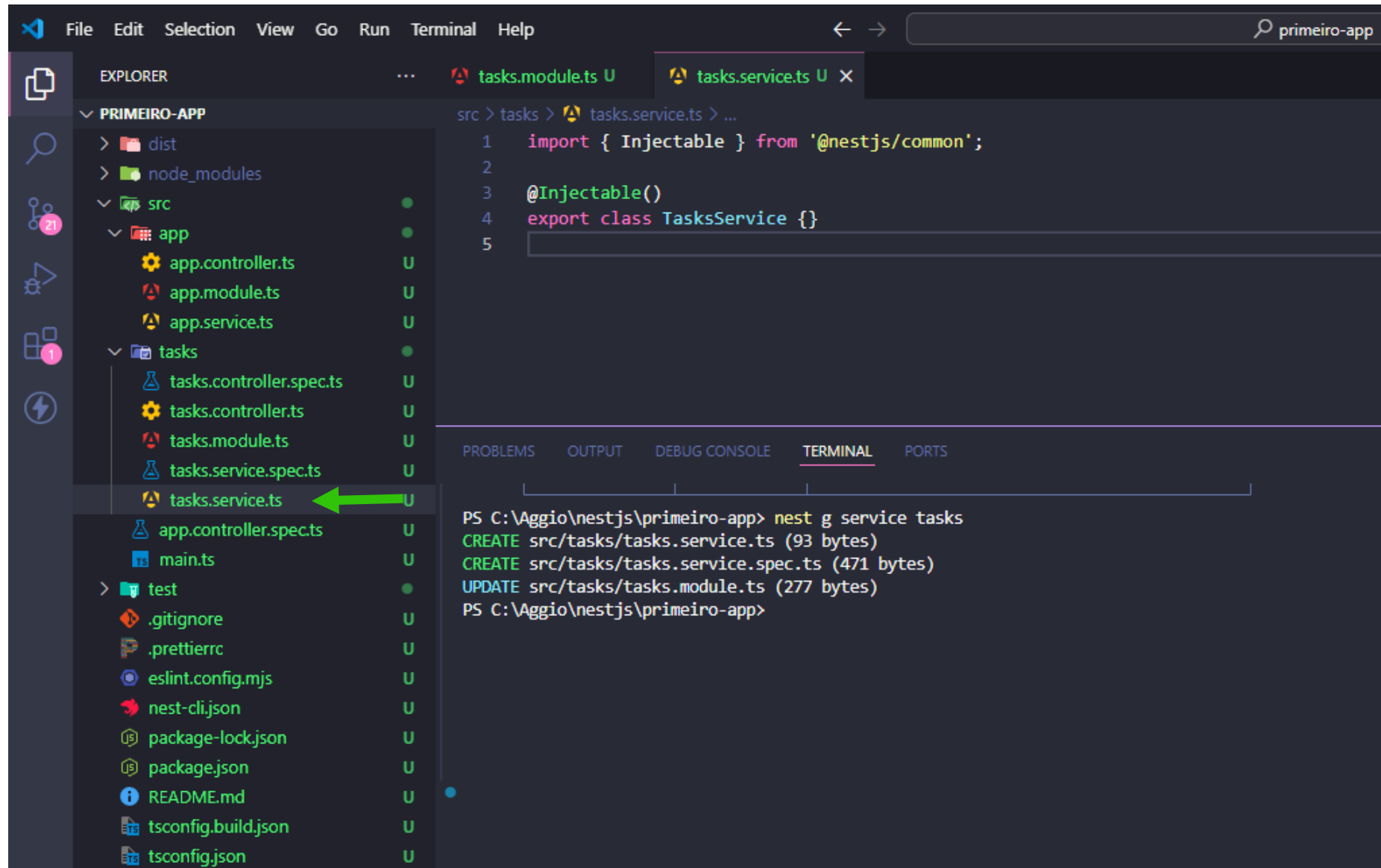
- new [n [options] [name]] Generate Nest application.
- build [options] [apps...] Build Nest application.
- start [options] [app] Run Nest application.
- info [i] Display Nest project details.
- add [options] <library> Adds support for an external library to your project.
- generate [g [options] <schematic> [name] [path]] Generate a Nest element.

Schematics available on @nestjs/schematics collection:

name	alias	description
application	application	Generate a new application workspace
class	cl	Generate a new class
configuration	config	Generate a CLI configuration file
controller	co	Generate a controller declaration
decorator	d	Generate a custom decorator
filter	f	Generate a filter declaration
gateway	ga	Generate a gateway declaration
guard	gu	Generate a guard declaration
interceptor	itc	Generate an interceptor declaration
interface	itf	Generate an interface
library	lib	Generate a new library within a monorepo
middleware	mi	Generate a middleware declaration
module	mo	Generate a module declaration
pipe	pi	Generate a pipe declaration
provider	pr	Generate a provider declaration
resolver	r	Generate a GraphQL resolver declaration
resource	res	Generate a new CRUD resource
service	s	Generate a service declaration
sub-app	app	Generate a new application within a monorepo

PS C:\Vaggio\nestjs\primeiro-app> nest g service tasks

Gerando Service / Provider



The screenshot shows the Visual Studio Code interface with the Explorer, Editor, and Terminal views. The Explorer view on the left shows the project structure for 'PRIMEIRO-APP'. The Editor view in the center shows the code for 'tasks.service.ts'. The Terminal view at the bottom shows the command 'nest g service tasks' and its output.

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - app.controller.ts
 - app.module.ts
 - app.service.ts
 - tasks
 - tasks.controller.spec.ts
 - tasks.controller.ts
 - tasks.module.ts
 - tasks.service.spec.ts
 - tasks.service.ts
 - app.controller.spec.ts
 - main.ts
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.build.json
 - tsconfig.json

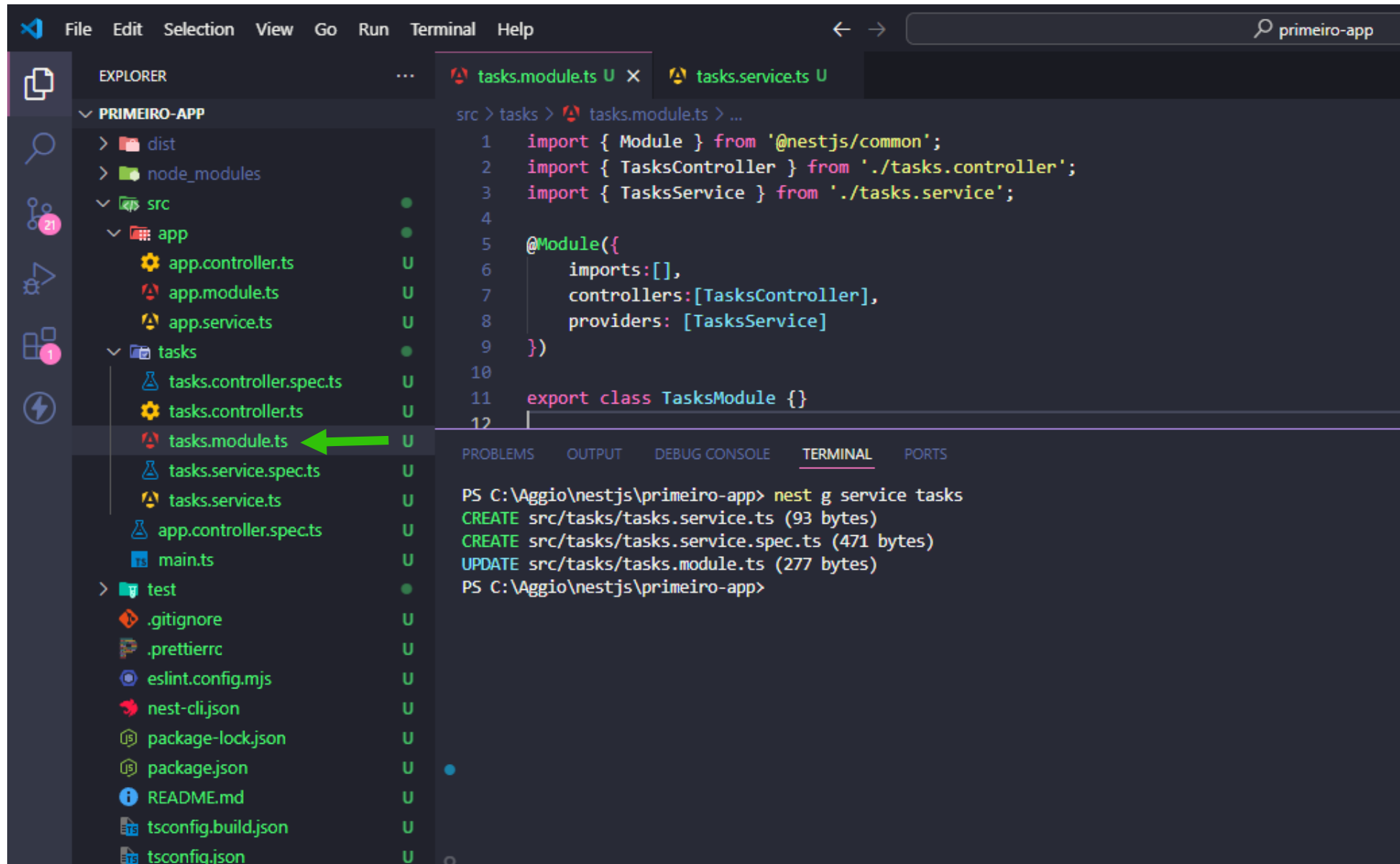
tasks.service.ts

```
1 import { Injectable } from '@nestjs/common';
2
3 @Injectable()
4 export class TasksService {}
5
```

TERMINAL

```
PS C:\Aggio\nestjs\primeiro-app> nest g service tasks
CREATE src/tasks/tasks.service.ts (93 bytes)
CREATE src/tasks/tasks.service.spec.ts (471 bytes)
UPDATE src/tasks/tasks.module.ts (277 bytes)
PS C:\Aggio\nestjs\primeiro-app>
```

Gerando Service / Provider



The screenshot shows the Visual Studio Code interface with the Explorer, Source Control, and Run and Debug views. The Explorer view shows the file structure of the 'PRIMEIRO-APP' project. The Source Control view shows the changes made to the 'tasks.module.ts' file. The Run and Debug view shows the output of the 'nest g service tasks' command.

EXPLORER

- PRIMEIRO-APP
 - dist
 - node_modules
 - src
 - app
 - app.controller.ts
 - app.module.ts
 - app.service.ts
 - tasks
 - tasks.controller.spec.ts
 - tasks.controller.ts
 - tasks.module.ts
 - tasks.service.spec.ts
 - tasks.service.ts
 - app.controller.spec.ts
 - main.ts
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.build.json
 - tsconfig.json

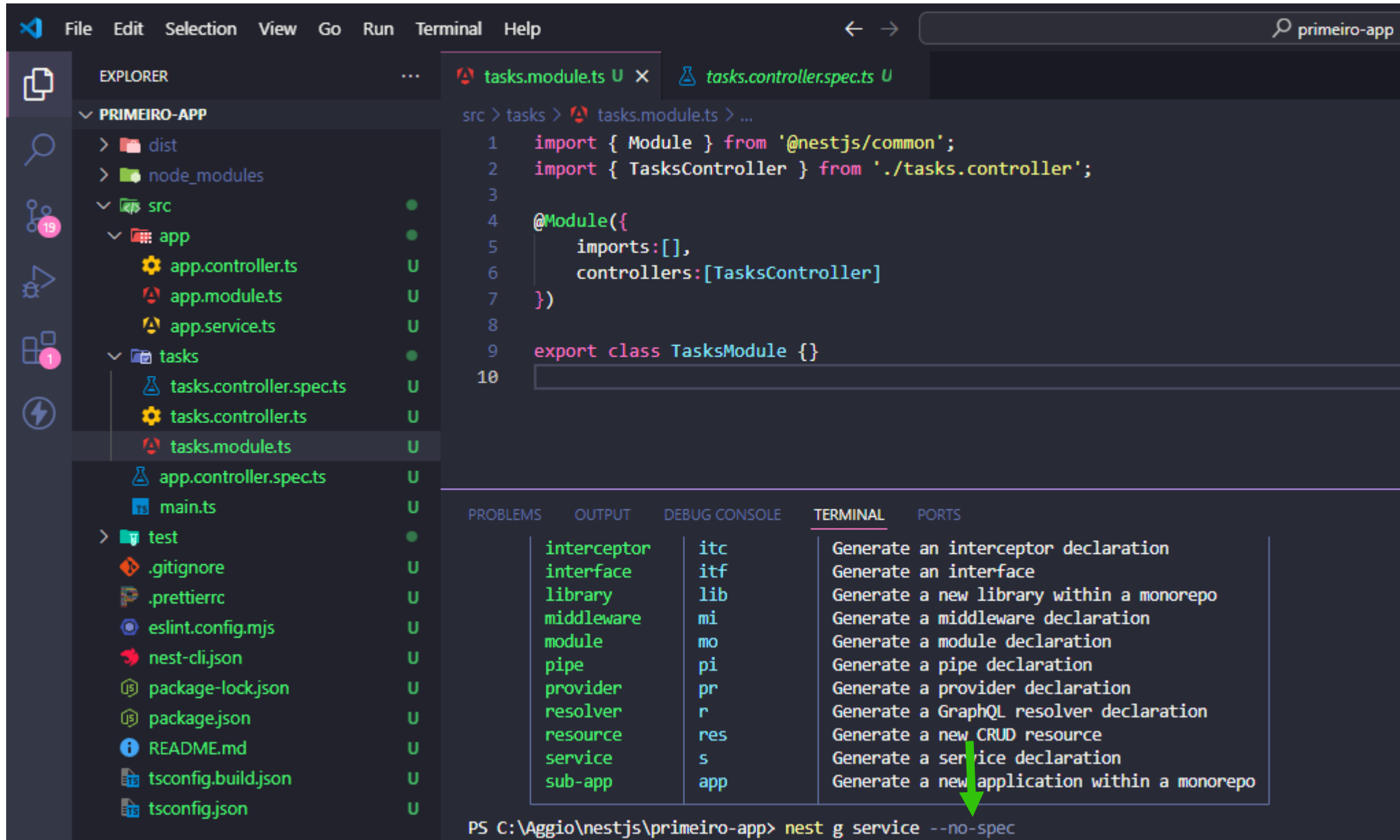
tasks.module.ts

```
1 import { Module } from '@nestjs/common';
2 import { TasksController } from './tasks.controller';
3 import { TasksService } from './tasks.service';
4
5 @Module({
6   imports: [],
7   controllers: [TasksController],
8   providers: [TasksService]
9 })
10
11 export class TasksModule {}
```

TERMINAL

```
PS C:\Aggio\nestjs\primeiro-app> nest g service tasks
CREATE src/tasks/tasks.service.ts (93 bytes)
CREATE src/tasks/tasks.service.spec.ts (471 bytes)
UPDATE src/tasks/tasks.module.ts (277 bytes)
PS C:\Aggio\nestjs\primeiro-app>
```

Gerando Service / Provider



The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** Displays the project structure for 'PRIMEIRO-APP'. The 'tasks' folder is expanded, showing 'tasks.module.ts' selected.
- EDITOR:** Shows the code for 'tasks.module.ts'. The code is as follows:


```
1 import { Module } from '@nestjs/common';
2 import { TasksController } from '../tasks.controller';
3
4 @Module({
5   imports:[],
6   controllers:[TasksController]
7 })
8
9 export class TasksModule {}
10
```
- TERMINAL:** Shows the command 'nest g service --no-spec' being executed. A green arrow points to the 'service' option in the list of available generators.

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL	PORTS
	interceptor	itc	Generate an interceptor declaration	
	interface	itf	Generate an interface	
	library	lib	Generate a new library within a monorepo	
	middleware	mi	Generate a middleware declaration	
	module	mo	Generate a module declaration	
	pipe	pi	Generate a pipe declaration	
	provider	pr	Generate a provider declaration	
	resolver	r	Generate a GraphQL resolver declaration	
	resource	res	Generate a new CRUD resource	
	service	s	Generate a service declaration	
	sub-app	app	Generate a new application within a monorepo	

PS C:\Aggio\nestjs\primeiro-app> nest g service --no-spec

Gerando Service / Provider

The screenshot displays the Visual Studio Code interface for a project named 'PRIMEIRO-APP'. The Explorer on the left shows the file structure, including 'src', 'app', and 'tasks' folders. The Editor shows the 'tasks.module.ts' file with the following code:

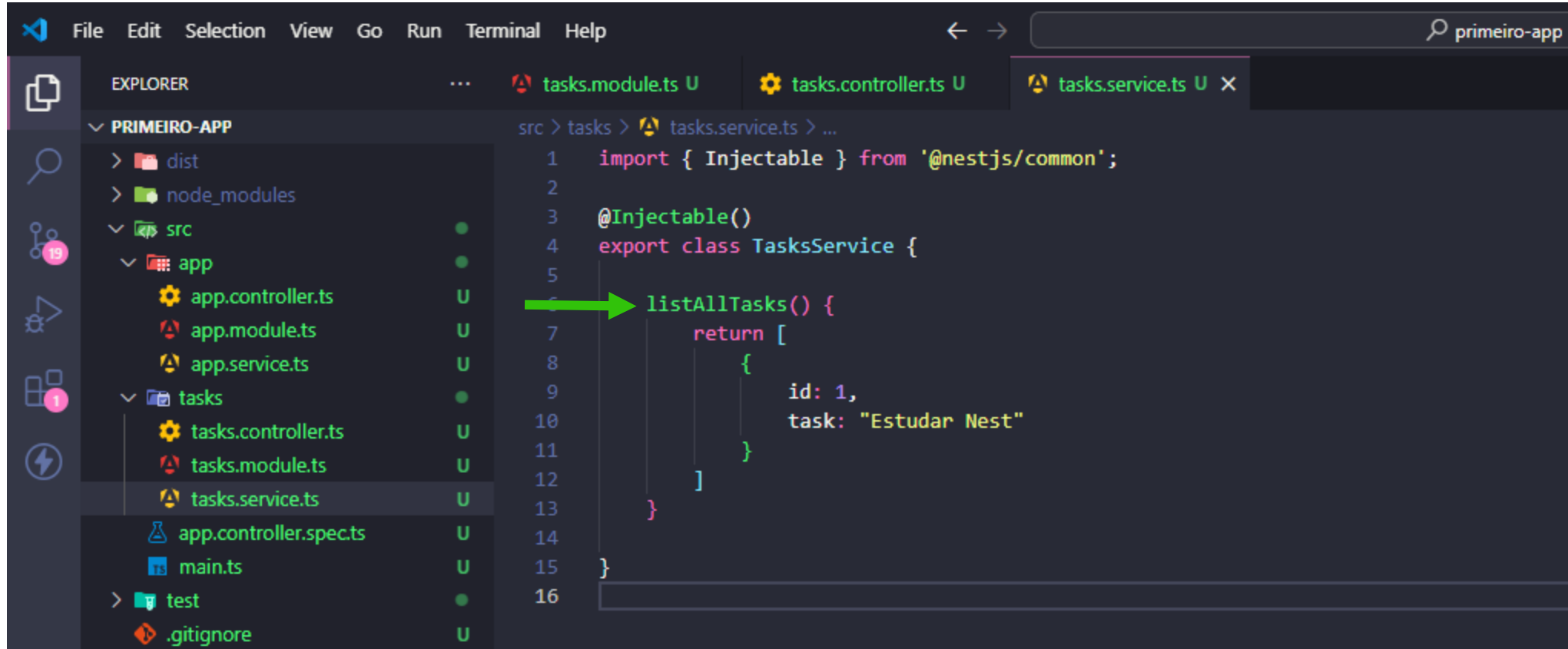
```
1 import { Module } from '@nestjs/common';
2 import { TasksController } from './tasks.controller';
3 import { TasksService } from './tasks.service';
4
5 @Module({
6   imports: [],
7   controllers: [TasksController],
8   providers: [TasksService]
9 })
10
11 export class TasksModule {}
```

The Terminal at the bottom shows the command 'nest g service tasks --no-spec' and its output:

```
PS C:\Aggio\nestjs\primeiro-app> nest g service tasks --no-spec
CREATE src/tasks/tasks.service.ts (93 bytes)
UPDATE src/tasks/tasks.module.ts (277 bytes)
```

A red arrow points from the text 'Sem gerar o arquivo de test' to the 'tasks.service.ts' file in the Explorer.

Gerando Service / Provider

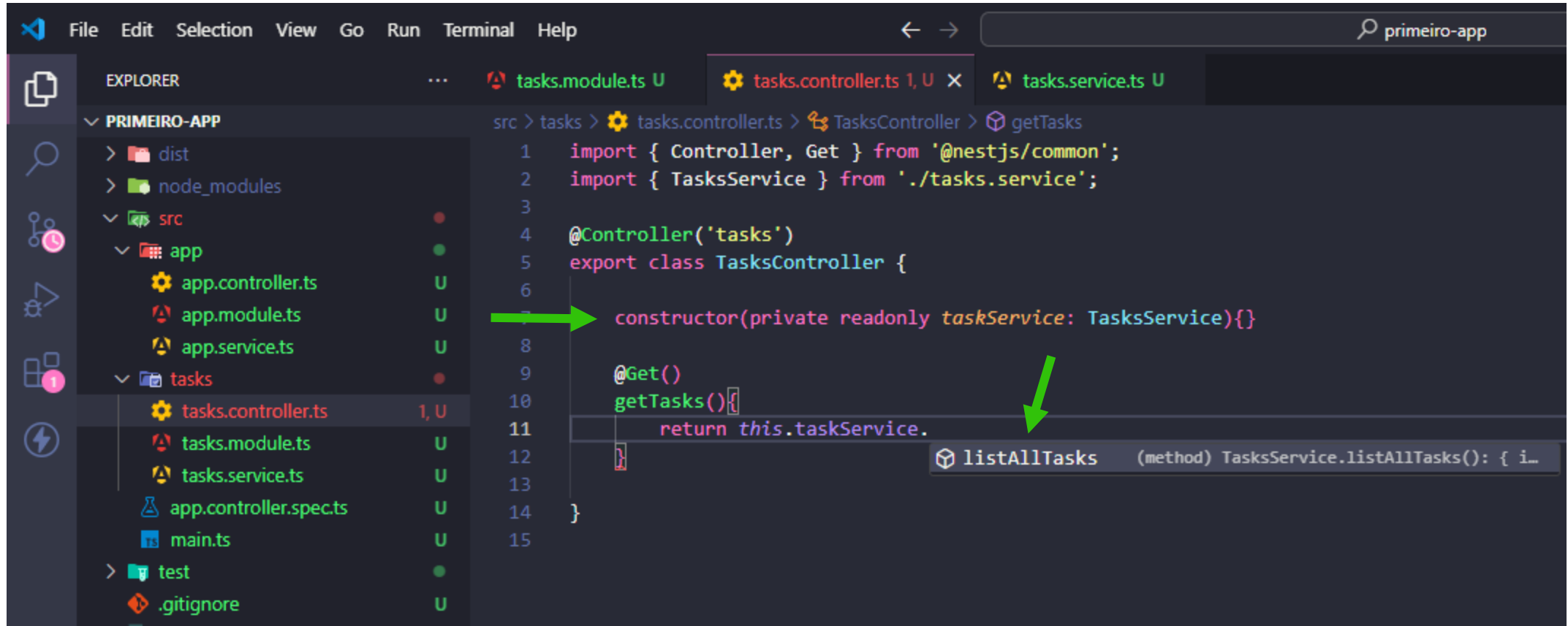


```
File Edit Selection View Go Run Terminal Help
primeiro-app

EXPLORER
PRIMEIRO-APP
  dist
  node_modules
  src
    app
      app.controller.ts
      app.module.ts
      app.service.ts
    tasks
      tasks.controller.ts
      tasks.module.ts
      tasks.service.ts
  app.controller.spec.ts
  main.ts
  test
  .gitignore

src > tasks > tasks.service.ts > ...
1 import { Injectable } from '@nestjs/common';
2
3 @Injectable()
4 export class TasksService {
5
6   listAllTasks() {
7     return [
8       {
9         id: 1,
10        task: "Estudar Nest"
11      }
12     ]
13   }
14 }
15
16
```

Gerando Service / Provider



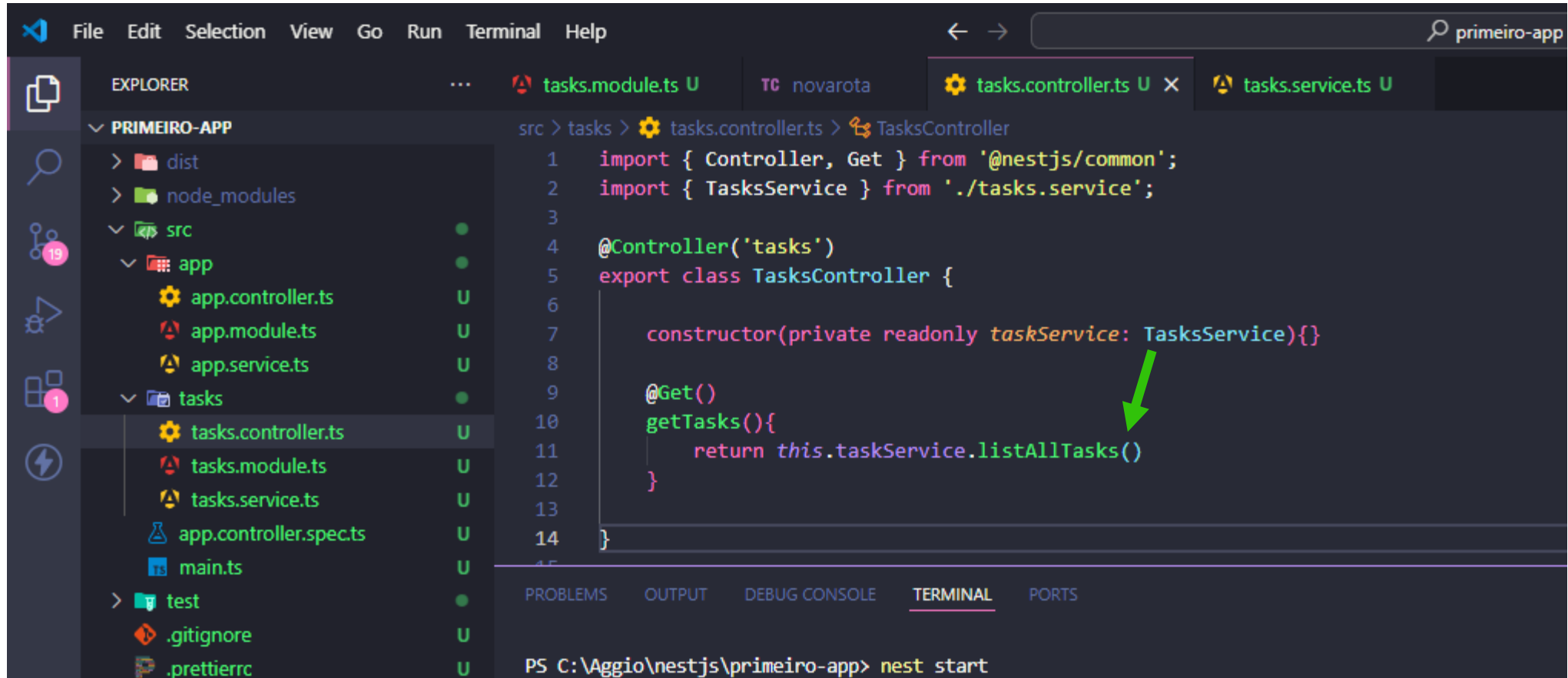
The screenshot shows the Visual Studio Code editor with the following components:

- EXPLORER:** Displays the project structure for 'PRIMEIRO-APP'. The 'tasks' folder is expanded, showing 'tasks.controller.ts' (1, U), 'tasks.module.ts' (U), and 'tasks.service.ts' (U).
- Editor:** Shows the code for 'tasks.controller.ts'. The file path is 'src > tasks > tasks.controller.ts > TasksController > getTasks'. The code is as follows:

```
1 import { Controller, Get } from '@nestjs/common';
2 import { TaskService } from '../tasks.service';
3
4 @Controller('tasks')
5 export class TasksController {
6
7   constructor(private readonly taskService: TaskService){}
8
9   @Get()
10  getTasks(){
11    return this.taskService.
12  }
13
14 }
```
- IntelliSense:** A tooltip is visible for the 'listAllTasks' property access on 'this.taskService', showing '(method) TaskService.listAllTasks(): { i...'

Two green arrows highlight key parts of the code: one points to the constructor parameter 'taskService' and the other points to the 'listAllTasks' property access in the 'getTasks' method.

Gerando Service / Provider



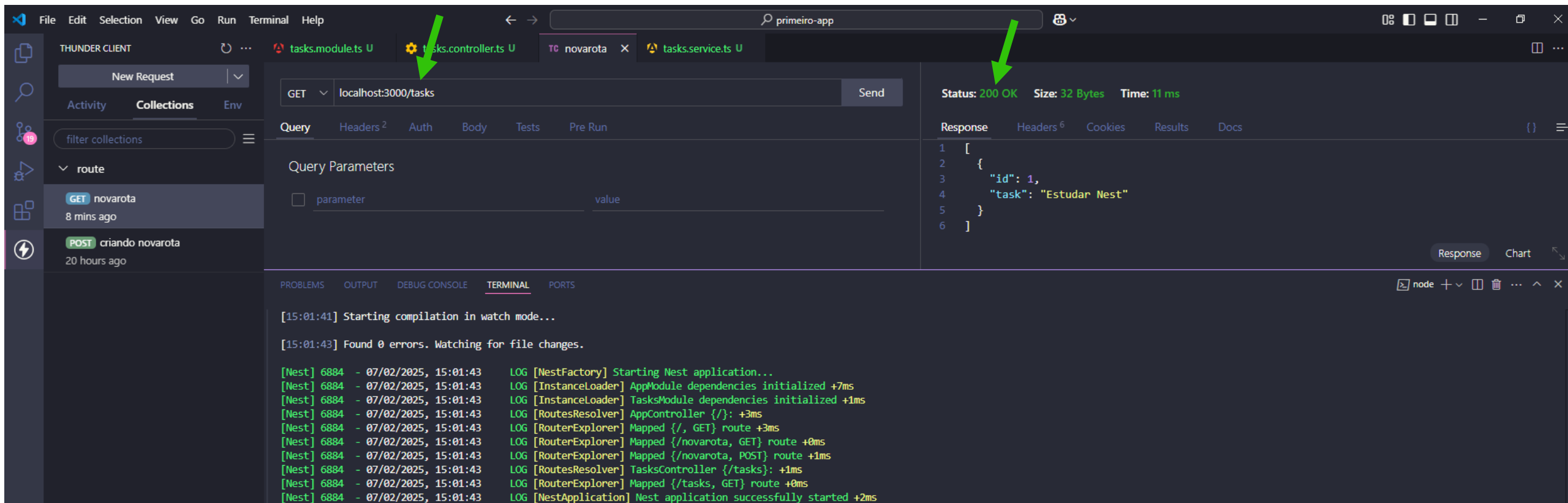
```
File Edit Selection View Go Run Terminal Help
primeiro-app

EXPLORER
PRIMEIRO-APP
  dist
  node_modules
  src
    app
      app.controller.ts
      app.module.ts
      app.service.ts
    tasks
      tasks.controller.ts
      tasks.module.ts
      tasks.service.ts
    app.controller.spec.ts
    main.ts
  test
  .gitignore
  .prettierrc

src > tasks > tasks.controller.ts > TasksController
1 import { Controller, Get } from '@nestjs/common';
2 import { TaskService } from './tasks.service';
3
4 @Controller('tasks')
5 export class TasksController {
6
7   constructor(private readonly taskService: TaskService){}
8
9   @Get()
10  getTasks(){
11    return this.taskService.listAllTasks()
12  }
13
14 }
15

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Aggio\nestjs\primeiro-app> nest start
```

Gerando Service / Provider



The screenshot displays the Thunder Client interface with a REST client request and response, and a terminal window showing the application logs.

Thunder Client Interface:

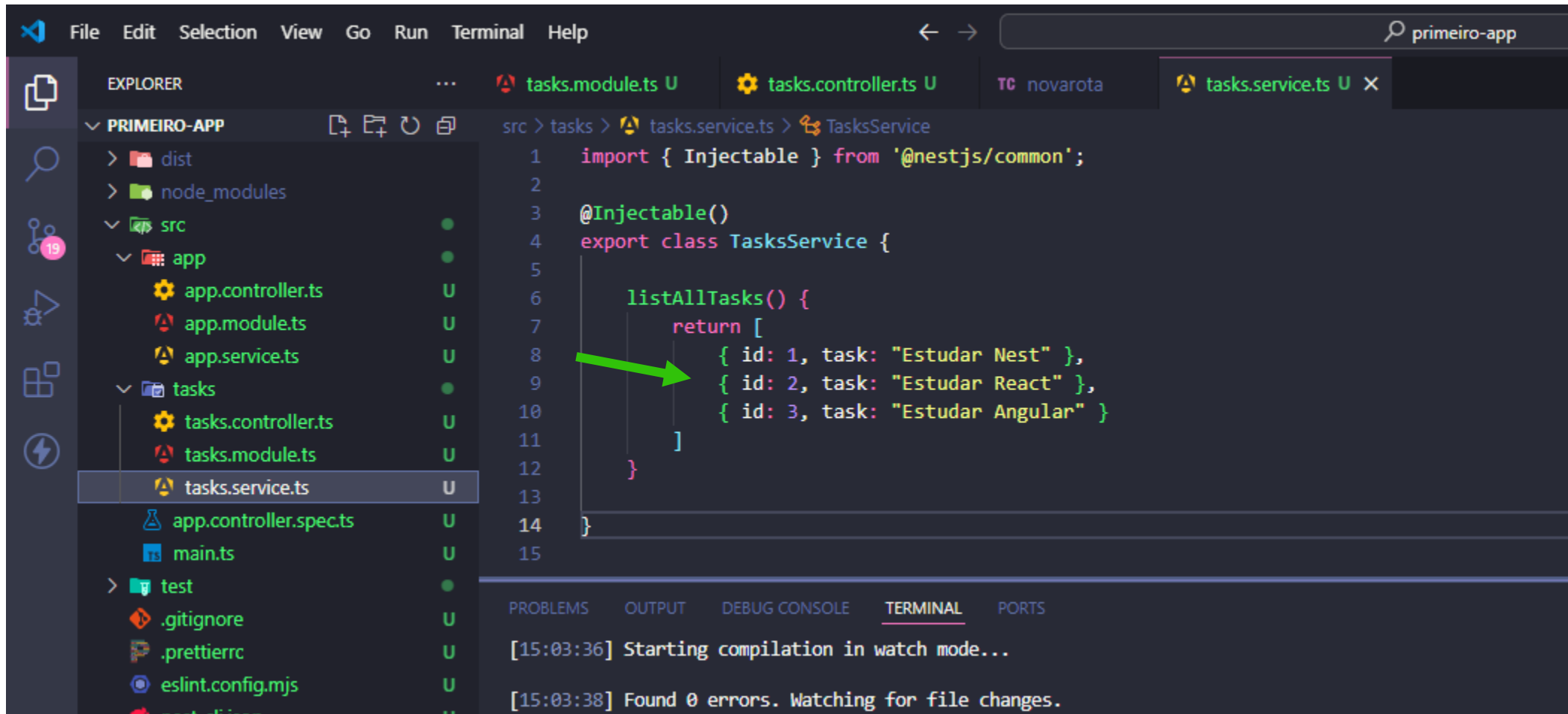
- Request:** GET localhost:3000/tasks
- Response:** Status: 200 OK, Size: 32 Bytes, Time: 11 ms
- Response Body:**

```
[{"id": 1, "task": "Estudar Nest"}]
```

Terminal Window:

```
[15:01:41] Starting compilation in watch mode...  
[15:01:43] Found 0 errors. Watching for file changes.  
  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [NestFactory] Starting Nest application...  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [InstanceLoader] AppModule dependencies initialized +7ms  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [InstanceLoader] TasksModule dependencies initialized +1ms  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [RoutesResolver] ApplicationController {/}: +3ms  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [RouterExplorer] Mapped {/, GET} route +3ms  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [RouterExplorer] Mapped {/novarota, GET} route +0ms  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [RouterExplorer] Mapped {/novarota, POST} route +1ms  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [RoutesResolver] TasksController {/tasks}: +1ms  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [RouterExplorer] Mapped {/tasks, GET} route +0ms  
[Nest] 6884 - 07/02/2025, 15:01:43 LOG [NestApplication] Nest application successfully started +2ms
```

Gerando Service / Provider



The screenshot shows the Visual Studio Code interface with the Explorer, Source Control, and Run and Debug views on the left. The Explorer view shows the project structure for 'PRIMEIRO-APP', with the 'tasks.service.ts' file selected. The Source Control view shows the file is staged. The Run and Debug view shows the file is being run. The main editor displays the code for 'tasks.service.ts', which implements the 'TaskService' interface. A green arrow points to the 'listAllTasks()' method, which returns an array of tasks. The terminal at the bottom shows the output of the compilation process.

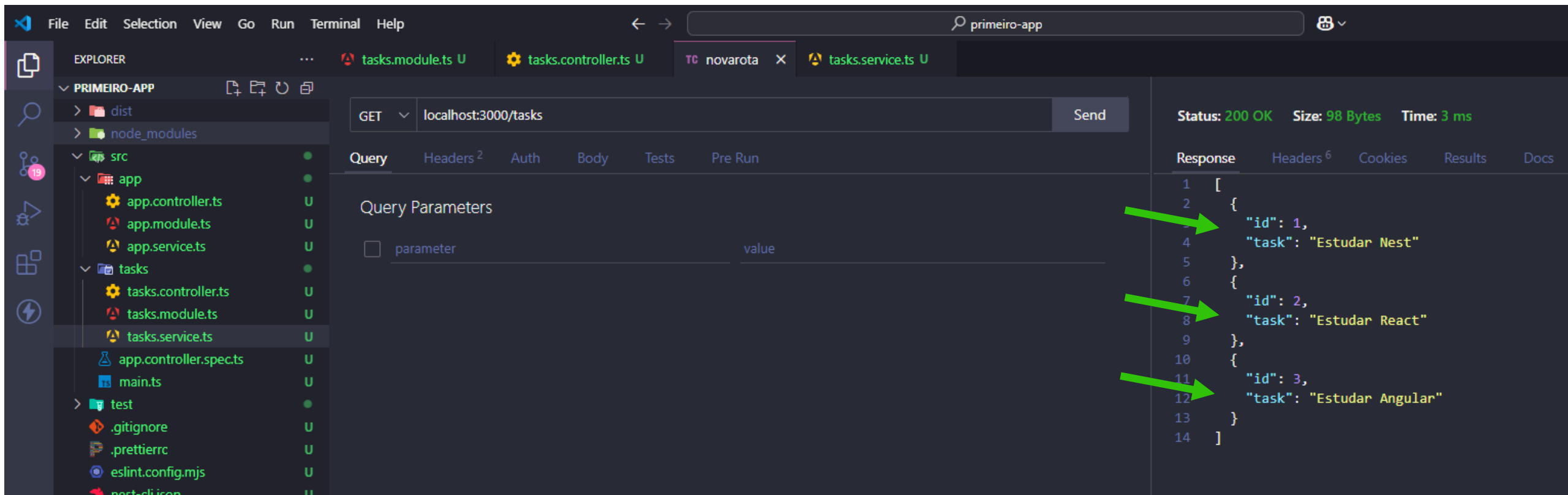
```
1 import { Injectable } from '@nestjs/common';
2
3 @Injectable()
4 export class TaskService {
5
6   listAllTasks() {
7     return [
8       { id: 1, task: "Estudar Nest" },
9       { id: 2, task: "Estudar React" },
10      { id: 3, task: "Estudar Angular" }
11    ]
12  }
13
14 }
15
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

[15:03:36] Starting compilation in watch mode...

[15:03:38] Found 0 errors. Watching for file changes.

Gerando Service / Provider



The screenshot shows the Visual Studio Code interface with a REST client request and response. The Explorer on the left shows the project structure for 'PRIMEIRO-APP', including files like 'app.controller.ts', 'app.module.ts', 'app.service.ts', 'tasks.controller.ts', 'tasks.module.ts', and 'tasks.service.ts'. The main editor displays a REST client request for 'GET localhost:3000/tasks'. The response is a JSON array of three tasks, each with an 'id' and a 'task' property. Three green arrows point from the response to the corresponding task objects in the JSON array.

File Edit Selection View Go Run Terminal Help

primeiro-app

EXPLORER

PRIMEIRO-APP

- dist
- node_modules
- src
 - app
 - app.controller.ts
 - app.module.ts
 - app.service.ts
 - tasks
 - tasks.controller.ts
 - tasks.module.ts
 - tasks.service.ts
 - app.controller.spec.ts
 - main.ts
 - test
 - .gitignore
 - .prettierrc
 - eslint.config.mjs
 - nest-cli.json

tasks.module.ts U tasks.controller.ts U TC novarota X tasks.service.ts U

GET localhost:3000/tasks Send

Query Headers² Auth Body Tests Pre Run

Query Parameters

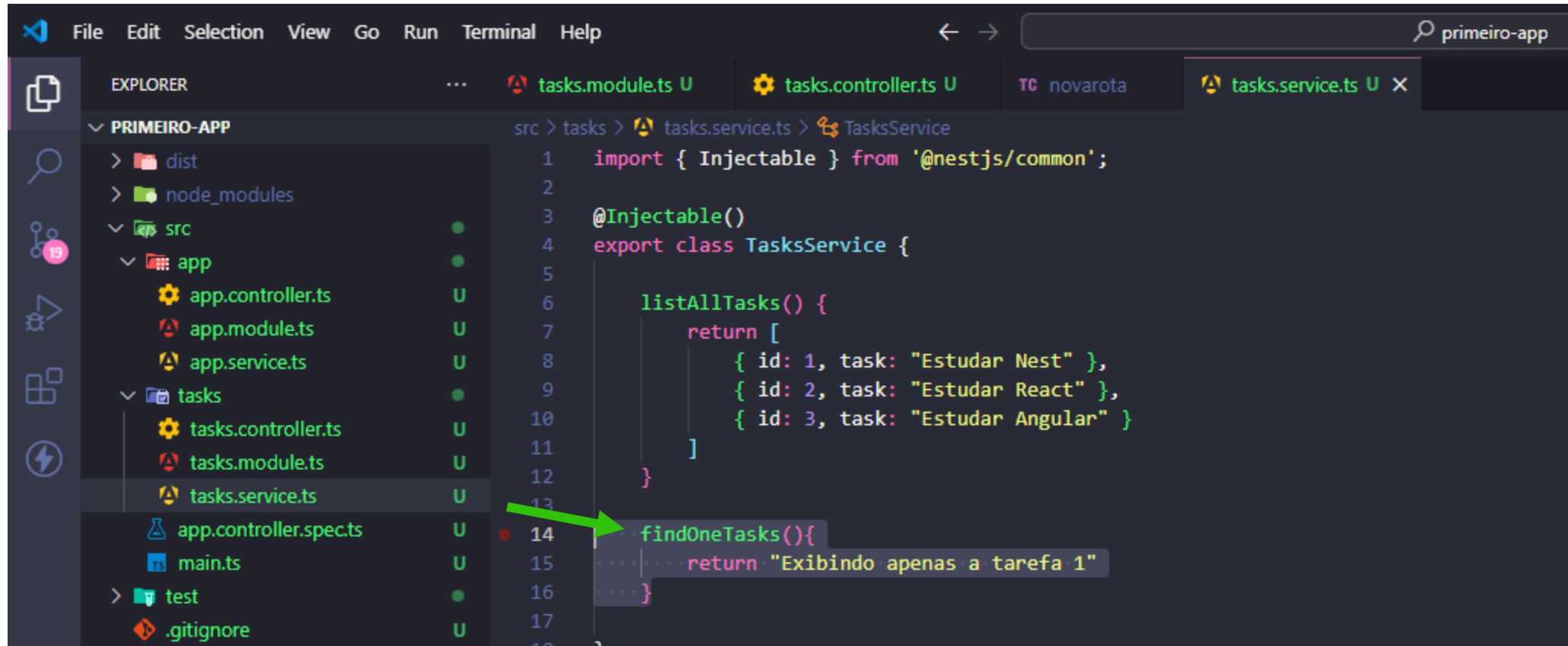
parameter value

Status: 200 OK Size: 98 Bytes Time: 3 ms

Response Headers⁶ Cookies Results Docs

```
1 [
2   {
3     "id": 1,
4     "task": "Estudar Nest"
5   },
6   {
7     "id": 2,
8     "task": "Estudar React"
9   },
10  {
11    "id": 3,
12    "task": "Estudar Angular"
13  }
14 ]
```

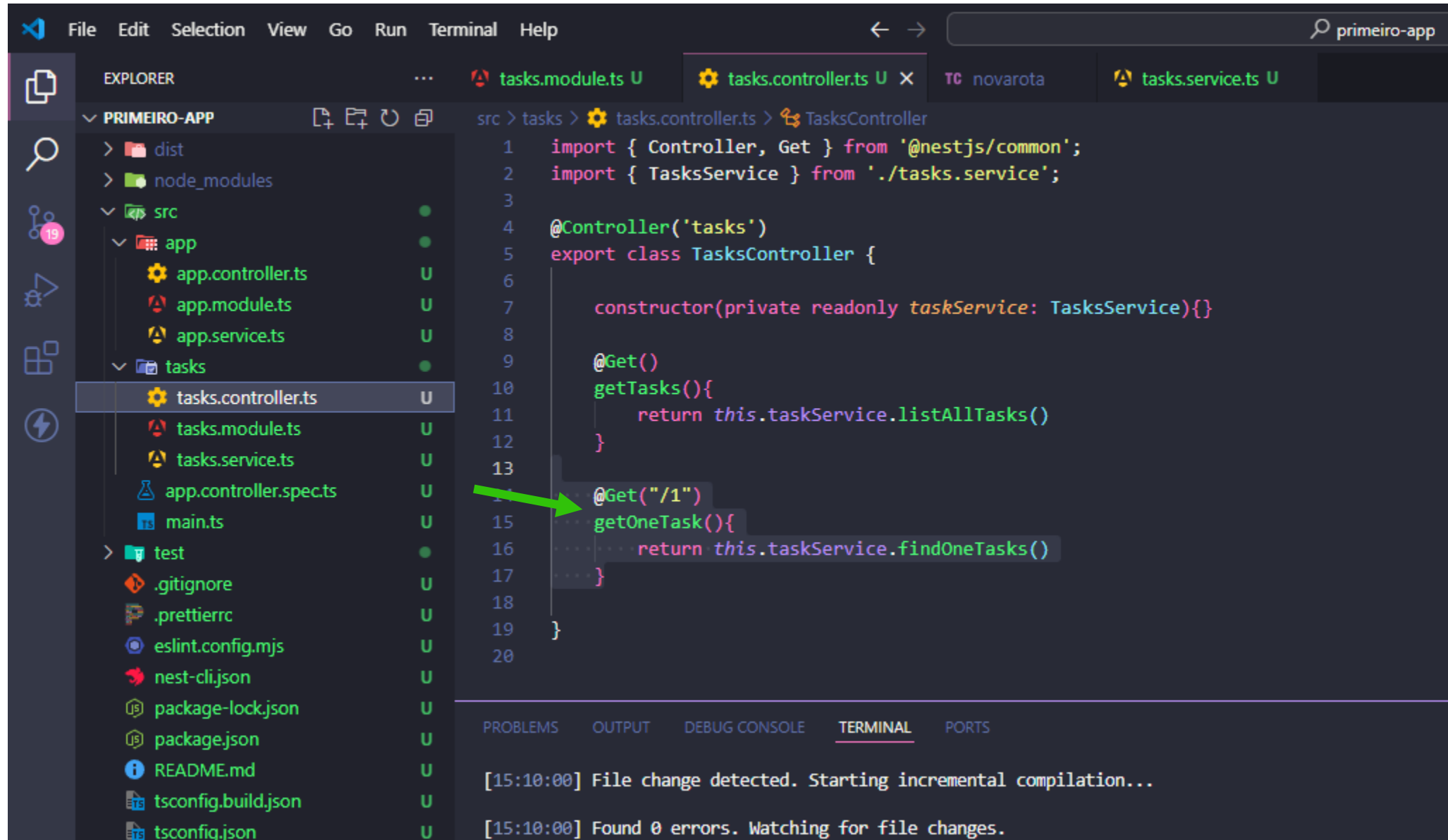
Gerando Service / Provider



The screenshot shows the Visual Studio Code editor interface. On the left, the Explorer sidebar displays the project structure for 'PRIMEIRO-APP'. The 'tasks' folder is expanded, and 'tasks.service.ts' is selected. The main editor area shows the code for 'tasks.service.ts'. The code includes an import for 'Injectable' from '@nestjs/common', a decorator '@Injectable()', and an export class 'TasksService'. The 'listAllTasks()' method returns an array of three task objects. The 'findOneTasks()' method is partially implemented, returning the string 'Exibindo apenas a tarefa 1'. A green arrow points to the 'findOneTasks()' method.

```
1 import { Injectable } from '@nestjs/common';
2
3 @Injectable()
4 export class TasksService {
5
6     listAllTasks() {
7         return [
8             { id: 1, task: "Estudar Nest" },
9             { id: 2, task: "Estudar React" },
10            { id: 3, task: "Estudar Angular" }
11        ]
12    }
13
14    findOneTasks(){
15        return "Exibindo apenas a tarefa 1"
16    }
17
18 }
```

Gerando Service / Provider



The screenshot shows the Visual Studio Code editor interface. On the left, the Explorer sidebar displays the project structure for 'PRIMEIRO-APP'. The 'tasks' folder is expanded, showing 'tasks.controller.ts' selected. The main editor area displays the code for 'tasks.controller.ts'. The code imports '@nestjs/common' and './tasks.service', defines a 'TasksController' class with a constructor that injects 'TasksService', and implements two methods: 'getTasks()' and 'getOneTask()'. A green arrow points to the '@Get("/1")' decorator on the 'getOneTask()' method. The bottom panel shows the 'TERMINAL' output with messages about incremental compilation and no errors found.

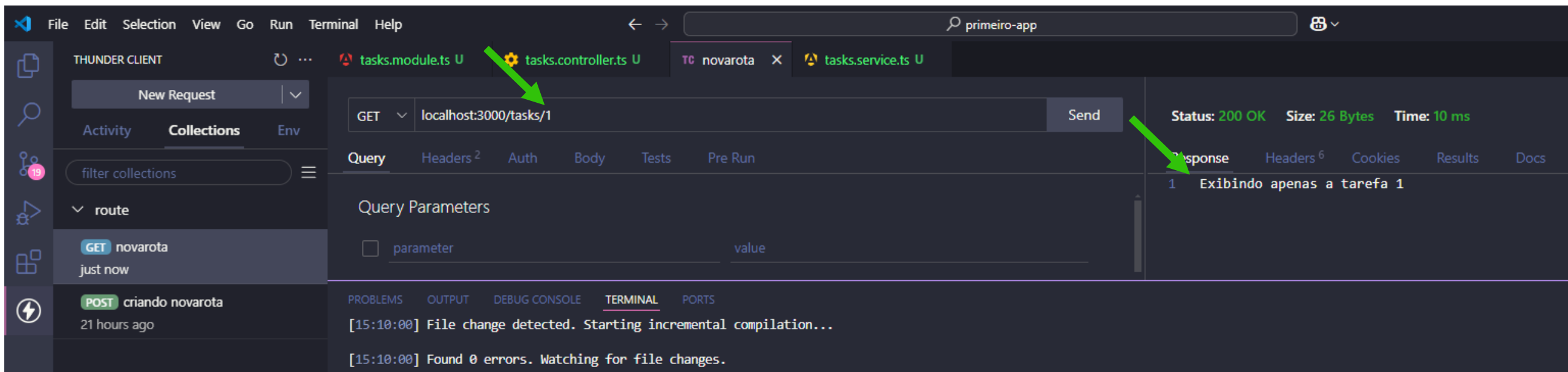
```
1 import { Controller, Get } from '@nestjs/common';
2 import { TasksService } from './tasks.service';
3
4 @Controller('tasks')
5 export class TasksController {
6
7     constructor(private readonly taskService: TasksService){}
8
9     @Get()
10    getTasks(){
11        return this.taskService.listAllTasks()
12    }
13
14    @Get("/1")
15    getOneTask(){
16        return this.taskService.findOneTasks()
17    }
18
19 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

[15:10:00] File change detected. Starting incremental compilation...

[15:10:00] Found 0 errors. Watching for file changes.

Gerando Service / Provider



The screenshot shows the VS Code interface with the Thunder Client extension. The file explorer on the left shows the project structure with files like `tasks.module.ts`, `tasks.controller.ts`, `novarota`, and `tasks.service.ts`. The Thunder Client interface is open, showing a REST client request for `GET localhost:3000/tasks/1`. The response is `200 OK` with a status of `200 OK`, size of `26 Bytes`, and time of `10 ms`. The response body is `Exibindo apenas a tarefa 1`. The terminal at the bottom shows the output of the application, including the message `[15:10:00] File change detected. Starting incremental compilation...` and `[15:10:00] Found 0 errors. Watching for file changes.`.

File Edit Selection View Go Run Terminal Help

THUNDER CLIENT

tasks.module.ts U tasks.controller.ts U TC novarota X tasks.service.ts U

New Request

Activity Collections Env

filter collections

route

GET novarota just now

POST criando novarota 21 hours ago

GET localhost:3000/tasks/1 Send

Status: 200 OK Size: 26 Bytes Time: 10 ms

Response Headers⁶ Cookies Results Docs

1 Exibindo apenas a tarefa 1

Query Parameters

parameter value

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

[15:10:00] File change detected. Starting incremental compilation...

[15:10:00] Found 0 errors. Watching for file changes.

Atividade

- 1) Crie as camadas: Guests, Users e Teachers**
- 2) Implemente sem utilizar a CLI: Module, Controller e Service**
- 3) Implemente o verbo Get, para listar tudo e por id**
- 4) Teste as rotas no Postman ou Thunder Client**
- 5) Refaça os itens de 1 a 4, utilizando a CLI**

CLI = Interface de Linha de Comando

Lembrete: nest g opçãoDoHelp nomeDaCamada