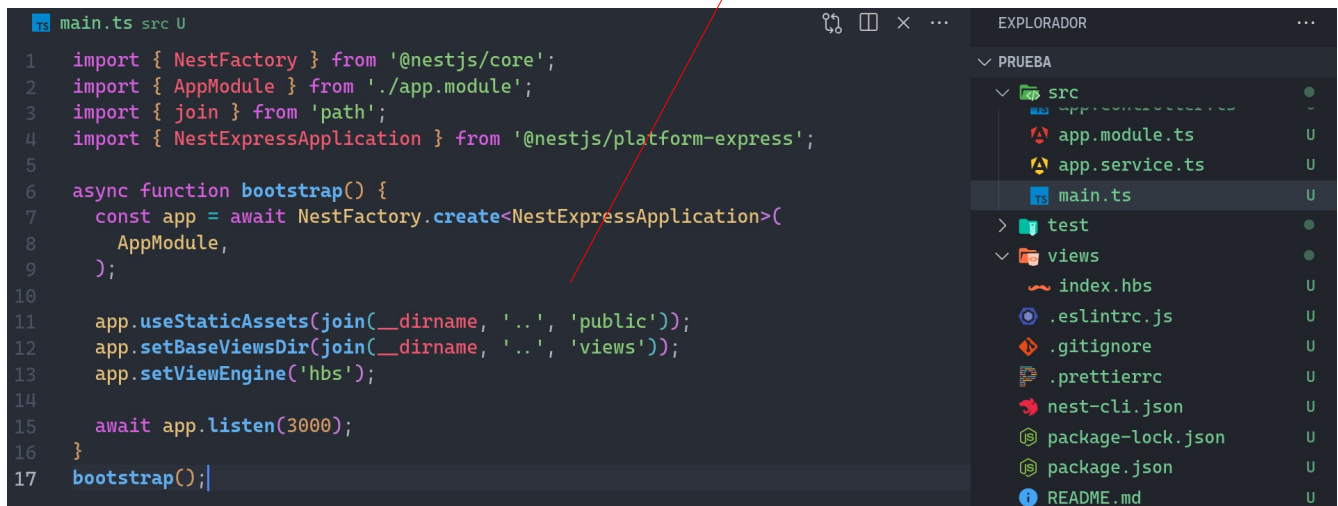


Acá se establecieron las rutas predeterminadas de las vistas



The screenshot shows the Visual Studio Code editor with the `main.ts` file open. The code defines a `bootstrap` function that creates a `NestExpressApplication` and sets up static assets, base views directory, and view engine. The Explorer sidebar on the right shows the project structure under the `PRUEBA` folder, including `src`, `test`, and `views` directories.

```
1 import { NestFactory } from '@nestjs/core';
2 import { AppModule } from './app.module';
3 import { join } from 'path';
4 import { NestExpressApplication } from '@nestjs/platform-express';
5
6 async function bootstrap() {
7   const app = await NestFactory.create<NestExpressApplication>(
8     AppModule,
9   );
10
11   app.useStaticAssets(join(__dirname, '..', 'public'));
12   app.setBaseViewsDir(join(__dirname, '..', 'views'));
13   app.setViewEngine('hbs');
14
15   await app.listen(3000);
16 }
17 bootstrap();
```

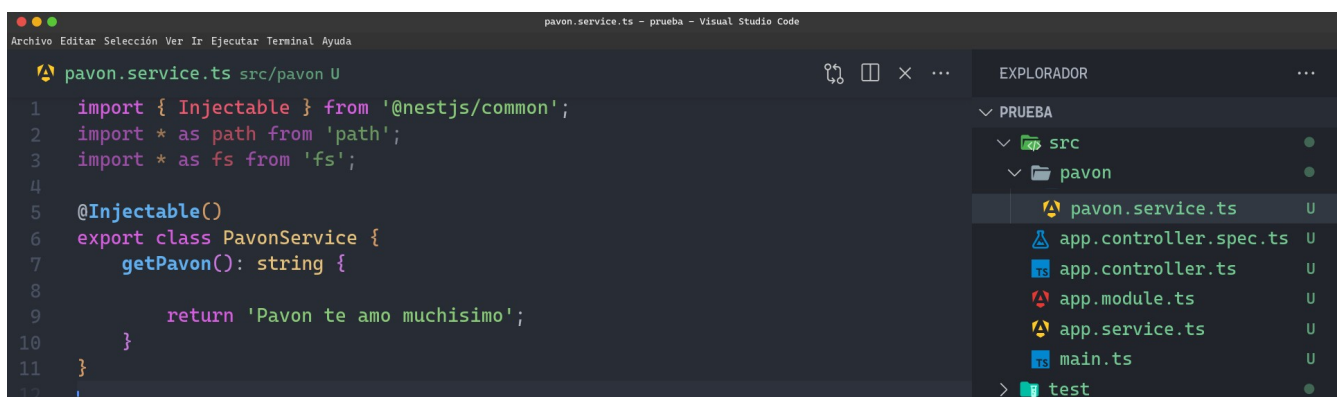
Micro servicio



The screenshot shows the Explorer sidebar with the `pavon` folder expanded. It lists several files related to the microservice, including controllers, services, and modules.

- `pavon.controller.spec.ts` U
- `pavon.controller.ts` U
- `pavon.module.ts` U
- `pavon.service.spec.ts` U
- `pavon.service.ts` U
- `app.controller.spec.ts` U
- `app.controller.ts` U
- `app.module.ts` U

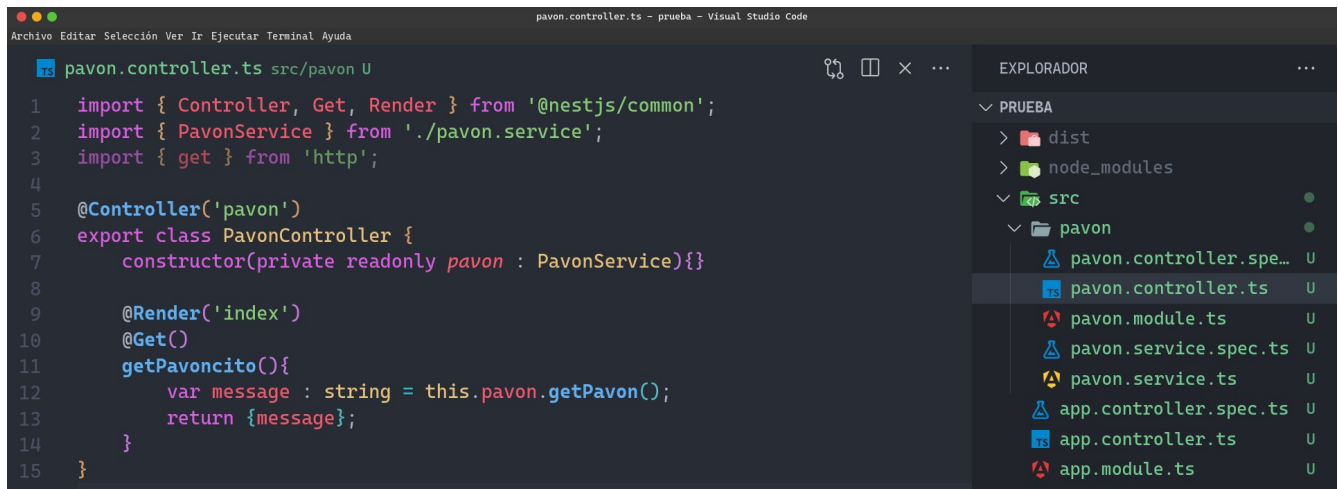
Se establecieron los servicios



The screenshot shows the Visual Studio Code editor with the `pavon.service.ts` file open. The code defines a `PavonService` class with a `getPavon` method that returns a string. The Explorer sidebar on the right shows the project structure, including the `pavon` folder.

```
1 import { Injectable } from '@nestjs/common';
2 import * as path from 'path';
3 import * as fs from 'fs';
4
5 @Injectable()
6 export class PavonService {
7   getPavon(): string {
8     return 'Pavon te amo muchisimo';
9   }
10 }
11
```

Se establecieron los servicios que retornan.

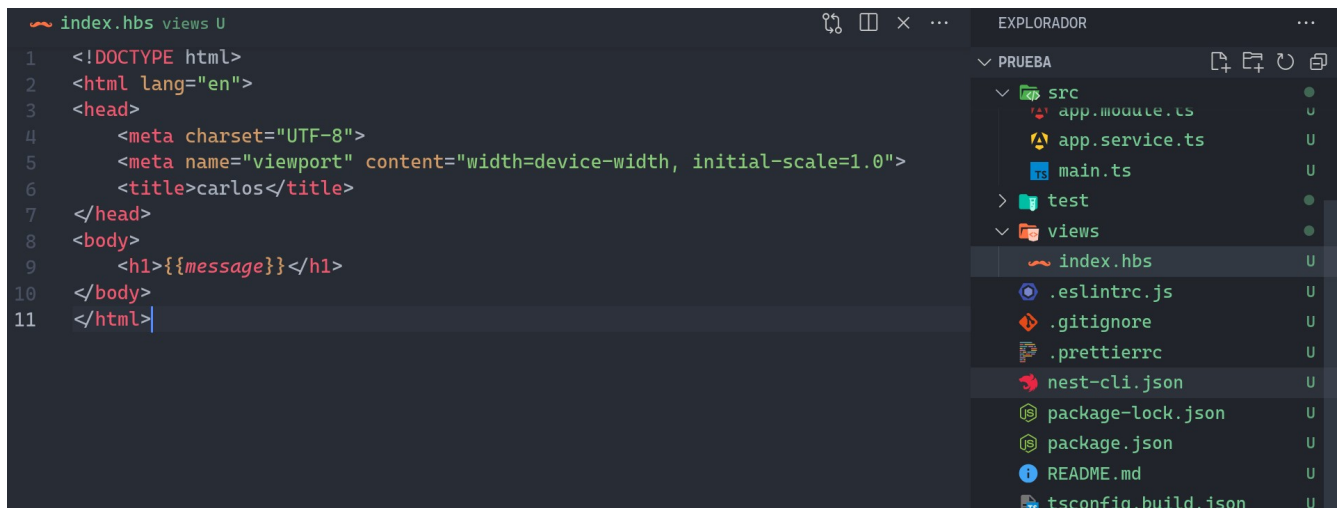


The screenshot shows the Visual Studio Code editor with a file named `pavon.controller.ts` in the `src/pavon` directory. The code implements a NestJS controller for a service named `PavonService`. It imports `Controller`, `Get`, and `Render` from `@nestjs/common`, and `PavonService` from `./pavon.service`. The controller has a constructor that takes a `PavonService` instance. It has a `Render` decorator for the `index` view and a `Get` decorator for the `getPavoncito` method. The `getPavoncito` method calls `this.pavon.getPavon()` and returns the result in a JSON object.

```
1 import { Controller, Get, Render } from '@nestjs/common';
2 import { PavonService } from '../pavon.service';
3 import { get } from 'http';
4
5 @Controller('pavon')
6 export class PavonController {
7   constructor(private readonly pavon : PavonService){}
8
9   @Render('index')
10  @Get()
11  getPavoncito(){
12    var message : string = this.pavon.getPavon();
13    return {message};
14  }
15 }
```

The Explorer sidebar on the right shows the project structure under `PRUEBA`, including `dist`, `node_modules`, `src`, and `pavon`. The `pavon` directory contains `pavon.controller.spec.ts`, `pavon.controller.ts`, `pavon.module.ts`, `pavon.service.spec.ts`, and `pavon.service.ts`.

Muestra el mensaje en el puerto y la ruta.



The screenshot shows the Visual Studio Code editor with a file named `index.hbs` in the `views` directory. The code is an HBS template that renders an HTML page. It includes a `DOCTYPE html` declaration, a `html` tag with `lang="en"`, a `head` tag with `meta` tags for `charset` and `viewport`, and a `title` tag with the text `carlos`. The `body` tag contains an `h1` tag with the message `{{message}}`.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>carlos</title>
7 </head>
8 <body>
9   <h1>{{message}}</h1>
10 </body>
11 </html>
```

The Explorer sidebar on the right shows the project structure under `PRUEBA`, including `src`, `test`, and `views`. The `src` directory contains `app.module.ts`, `app.service.ts`, and `main.ts`. The `views` directory contains `index.hbs`.

