

Universidad Autónoma de Chiapas UNACH

Facultad:

Contaduría y Administración, Campus I

Licenciatura:

Ingeniería en Desarrollo y Tecnologías de Software

Unidad de Aprendizaje:

Taller de desarrollo 4

Docente:

D.s.c. Luis Gutiérrez Alfaro

Alumno:

Lozano Monjarás David José

Grado y Grupo:

6° "M"

Matricula:

A210116

Actividad:

Pasos para realizar el examen

Tuxtla Gutiérrez, Chiapas

Fecha: 20 – Febrero – 2024

COMANDOS TALLER DESARROLLO 4

PowerShell:

Se ejecuta como administrador.

Por si no te deja poner comandos, se pone esto (esto es un permiso):

```
Set-ExecutionPolicy Bypass -Scope Process -Force;  
[System.Net.ServicePointManager]::SecurityProtocol =  
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object  
System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1  
'))
```

Después ve a la carpeta donde crearas el proyecto: ejemplo: `cd C:\pwebtaller4\taller4\api-taller>`

Y colocamos los siguientes comandos:

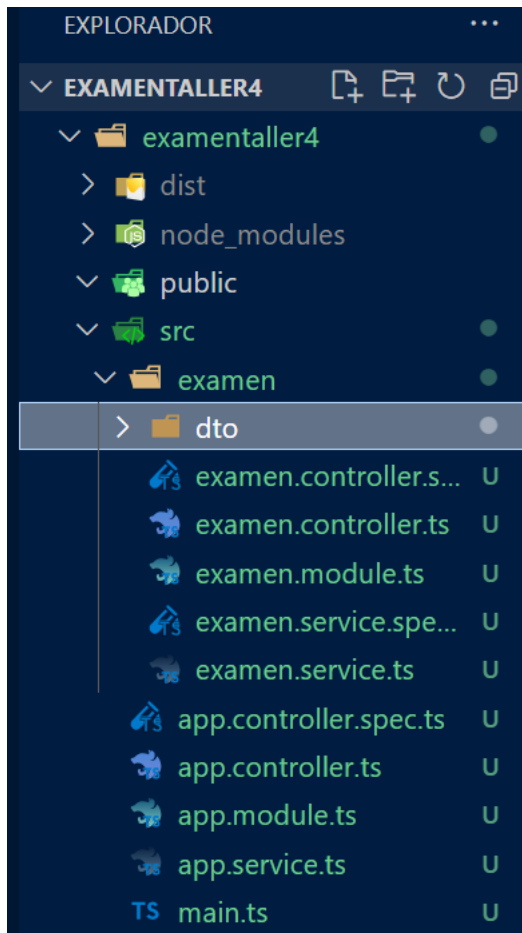
- `Cd ..`
- `Cd ..`
- `Cd .\nombrecarpetadelproyecto`
- `nest new nombre del proyecto`
seleccionar npm
- `code .`

Después de crear el archivo abrir la terminal y colocar los siguientes comandos en la terminal de visual studio code

- `Npm run build`
- `Npm run start:dev`

Creamos la carpeta dto y creamos las carpetas

- `Nest g co nombrepryecto`
- `Nest g s nombrepryecto`
- `Nest g mo nombrepryecto`



Luego siguiendo los pasos del MVC nestjs colocamos el comando para iniciar las modificaciones

```
$ npm install --save hbs
```

El siguiente paso es modificar el main.ts

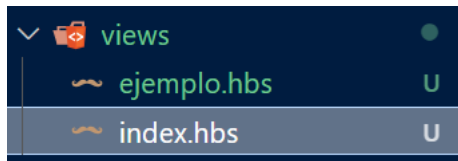
```
import { NestFactory } from '@nestjs/core';
import { NestExpressApplication } from '@nestjs/platform-express';
import { join } from 'path';
import { AppModule } from './app.module';

async function bootstrap() {
  const app = await NestFactory.create<NestExpressApplication>(
    AppModule,
  );

  app.useStaticAssets(join(__dirname, '..', 'public'));
  app.setBaseViewsDir(join(__dirname, '..', 'views'));
  app.setViewEngine('hbs');

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

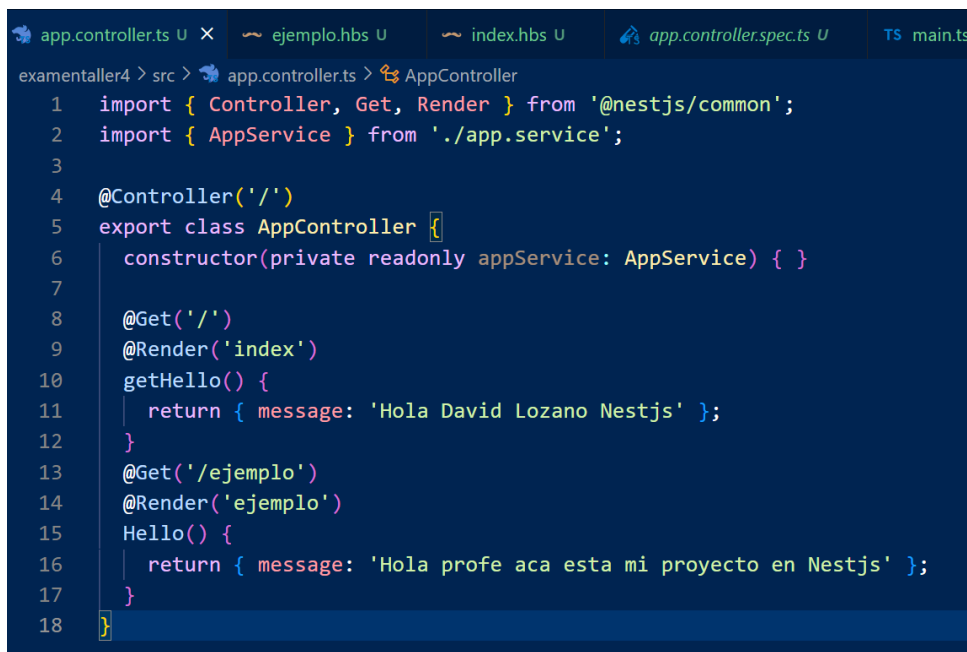
Y creamos la carpeta llamada views y en ella creamos el archivo index.hbs



Dentro del archivo colocamos este código

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>App</title>
  </head>
  <body>
    {{ message }}
  </body>
</html>
```

Luego modificamos nuestra carpeta app.controller.ts

A screenshot of a code editor showing the 'app.controller.ts' file. The code is written in TypeScript and defines an 'AppController' class. The class has a constructor that takes an 'AppService' as a dependency. It also has two methods: 'getHello()' and 'Hello()'. The 'getHello()' method returns a JSON object with a 'message' property. The 'Hello()' method also returns a JSON object with a 'message' property. The code is as follows:

```
1 import { Controller, Get, Render } 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   @Render('index')
10  getHello() {
11    return { message: 'Hola David Lozano Nestjs' };
12  }
13
14   @Get('/ejemplo')
15   @Render('ejemplo')
16   Hello() {
17     return { message: 'Hola profe aca esta mi proyecto en Nestjs' };
18   }
19 }
```

Para insertar imágenes a nuestro index.hbs tenemos que colocar la dirección URL de nuestra imagen y dejar nuestro código de la siguiente manera

```
</style>
</head>

<body>
  <h1>
    Hola a todos, porfin pase el examen de taller 4
  </h1>
  <div>
    
  </div>
</body>
</html>
```

Y para Correr nuestro servidor = npm run start:dev