



Universidad Autónoma de Chiapas

Licenciatura en Ingeniería en Desarrollo y Tecnología de Software, Campus 1



Tuxtla Gutiérrez Chiapas, 18 de Abril del 2024

Nombre del Docente:

GUTIERREZ ALFARO LUIS, DR

Nombre de la Materia:

Taller de Desarrollo 4

Nombre del Alumno:

Clemente López Jasson Jared

Semestre y Grupo:

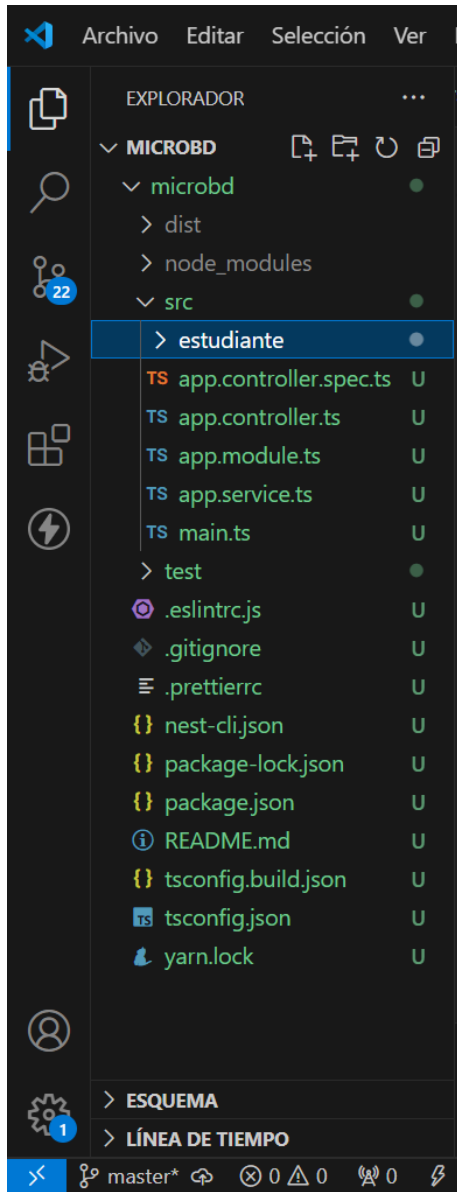
6-M

Actividad 3.1 MicroServicio con BD

(link de github)

https://github.com/Jasson2003/Actividad-3_1-MicroServicio-con-BD.git

Cree mi nest llamado “MICROBD”



Dentro de mi archivo instale:

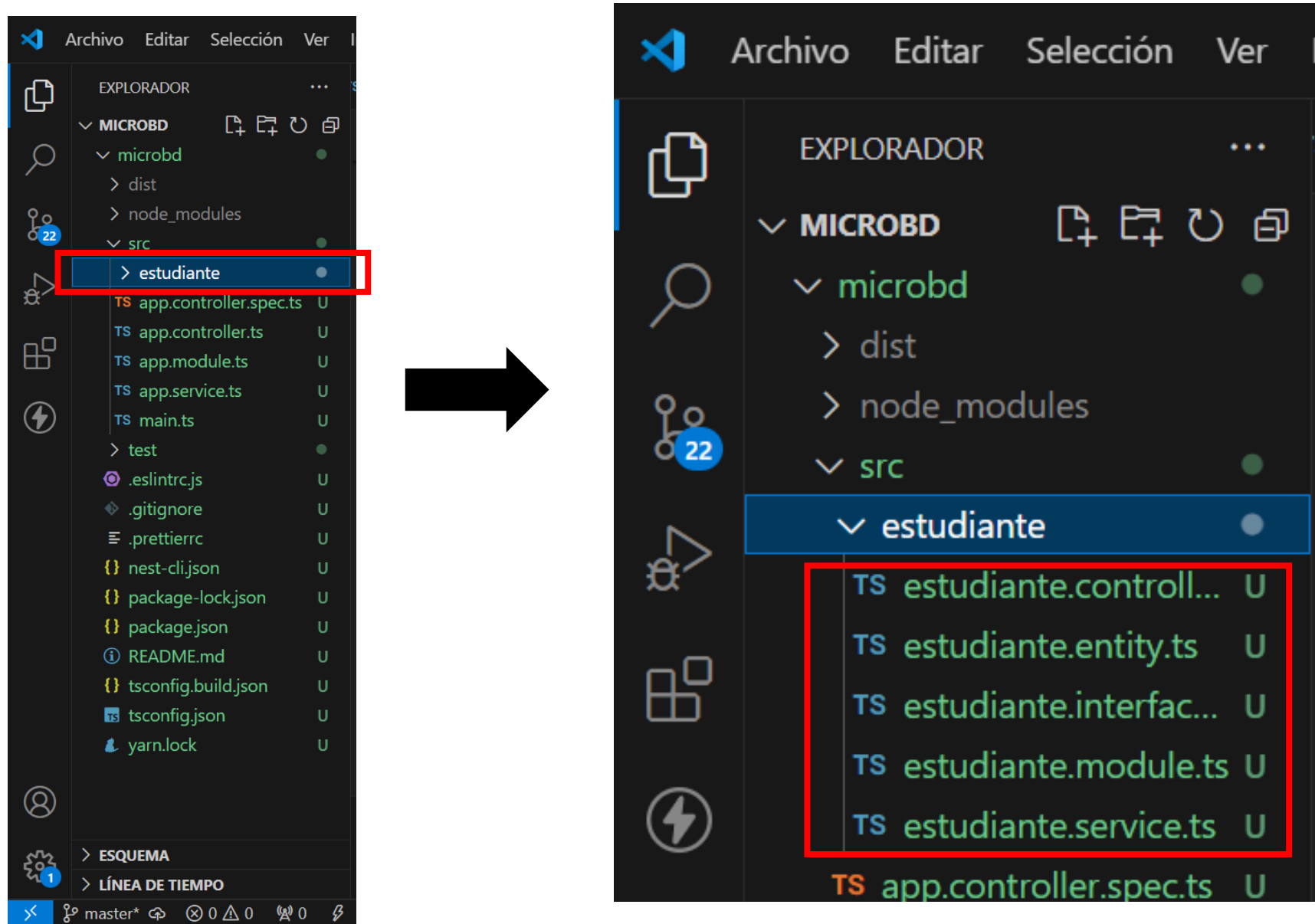
```
PS C:\Users\jasso\OneDrive\Documentos\MicroBD\microbd: npm install --save @nestjs/typeorm mysql2 typeorm

added 34 packages, and audited 749 packages in 19s

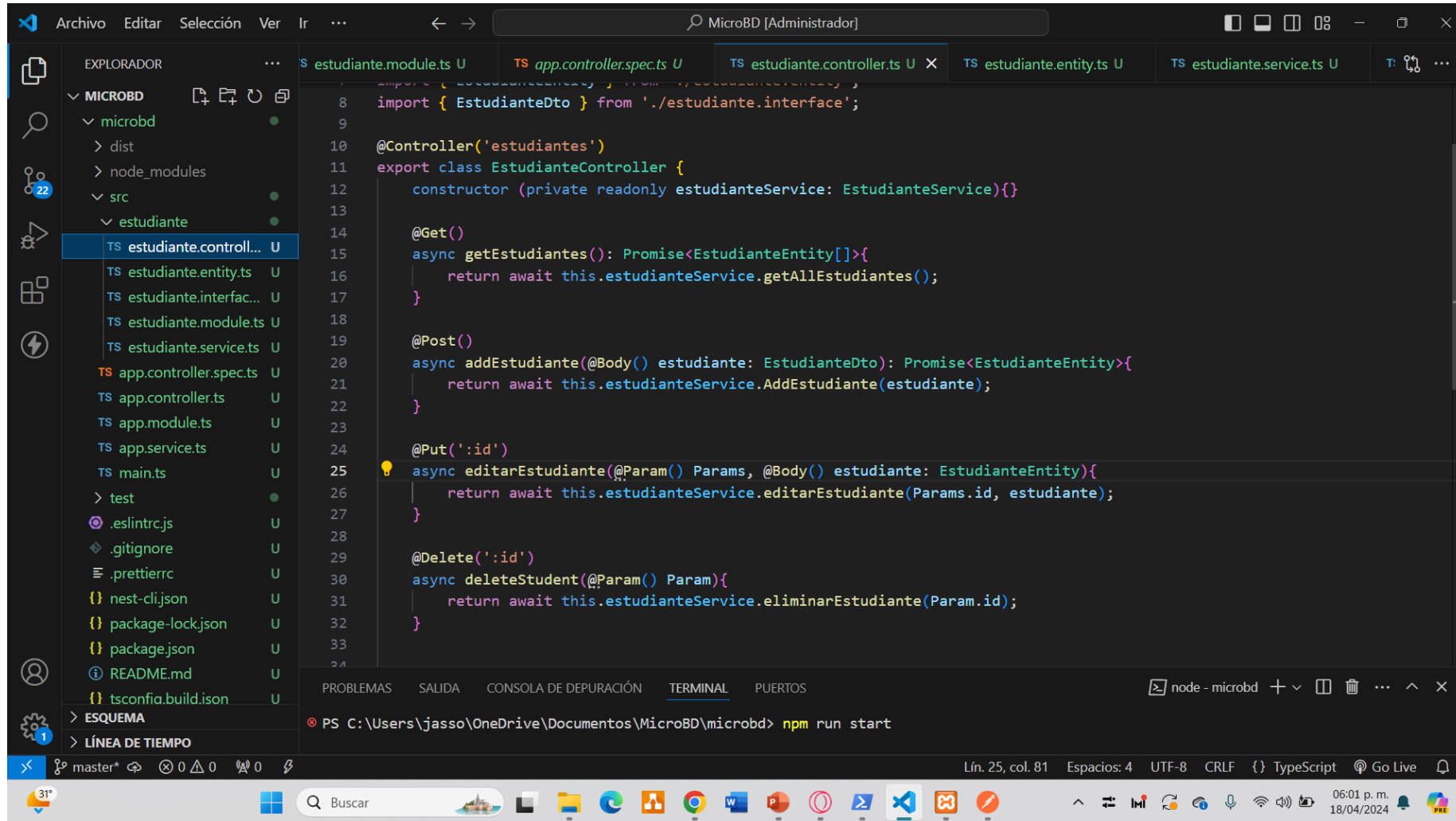
121 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\jasso\OneDrive\Documentos\MicroBD\microbd: yarn install
yarn install v1.22.19
info No lockfile found.
```

Cree una carpeta llamada “estudiante” ahí voy a crear archivos .ts



Se crea un controlador en Nest.js para manejar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) relacionadas con la entidad "Estudiante".



The screenshot shows the Visual Studio Code editor with a project named 'MicroBD'. The file explorer on the left shows the project structure, including a 'src' directory with an 'estudiante' subdirectory. The 'estudiante.controller.ts' file is open in the editor, showing the implementation of a Nest.js controller. The controller is named 'EstudianteController' and is associated with the 'estudiantes' route. It implements four CRUD operations: 'getEstudiantes' (GET), 'addEstudiante' (POST), 'editarEstudiante' (PUT), and 'deleteStudent' (DELETE). The controller uses the 'EstudianteService' to perform these operations. The terminal at the bottom shows the command 'npm run start' being executed.

```
import { EstudianteEntity } from './estudiante.entity';
import { EstudianteDto } from './estudiante.interface';

@Controller('estudiantes')
export class EstudianteController {
  constructor (private readonly estudianteService: EstudianteService){}

  @Get()
  async getEstudiantes(): Promise<EstudianteEntity[]>{
    return await this.estudianteService.getAllEstudiantes();
  }

  @Post()
  async addEstudiante(@Body() estudiante: EstudianteDto): Promise<EstudianteEntity>{
    return await this.estudianteService.AddEstudiante(estudiante);
  }

  @Put('/:id')
  async editarEstudiante(@Param() Params, @Body() estudiante: EstudianteEntity){
    return await this.estudianteService.editarEstudiante(Params.id, estudiante);
  }

  @Delete('/:id')
  async deleteStudent(@Param() Param){
    return await this.estudianteService.eliminarEstudiante(Param.id);
  }
}
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

PS C:\Users\jasso\OneDrive\Documentos\MicroBD\microbd> npm run start

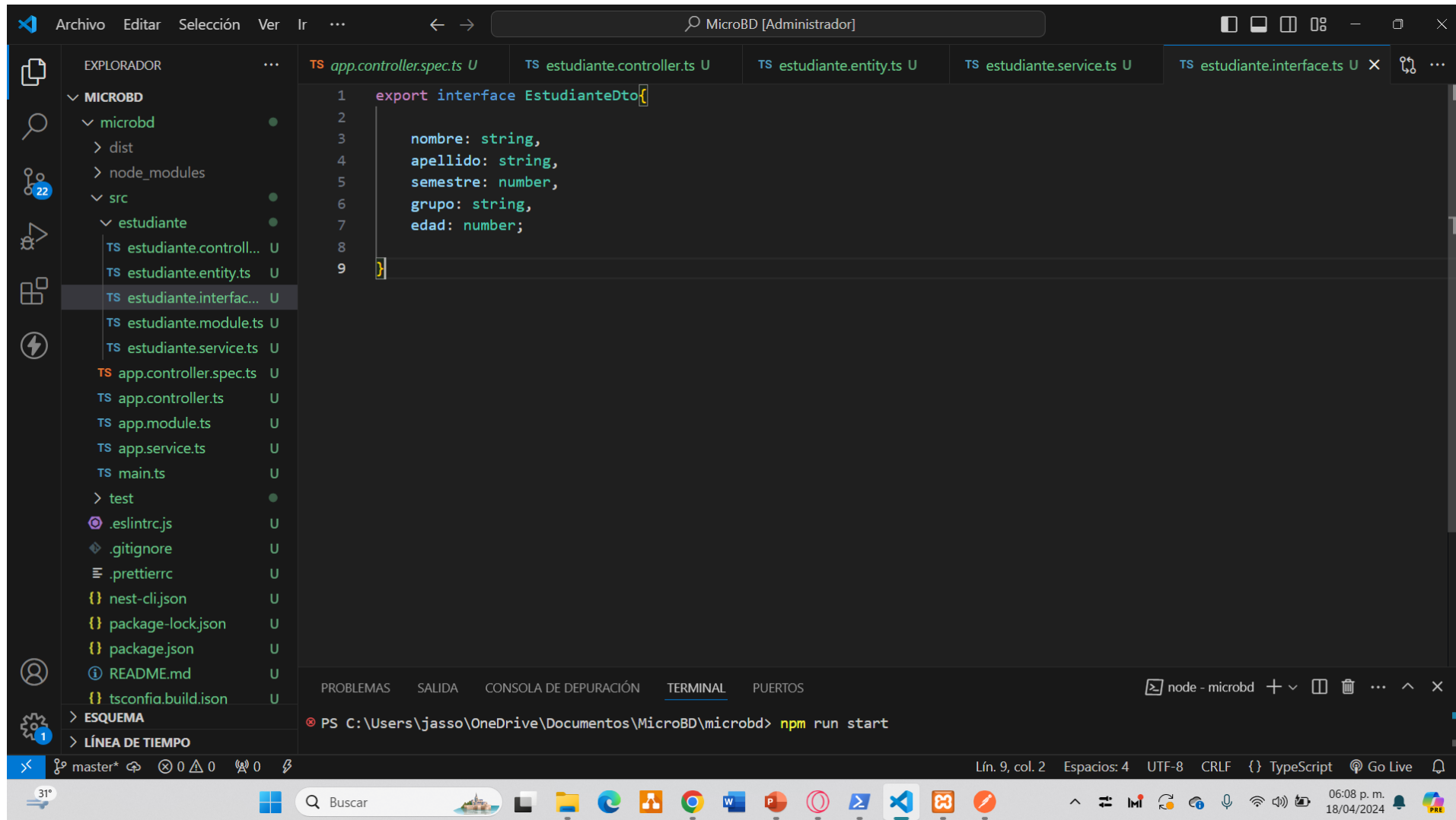
Lín. 25, col. 81 Espacios: 4 UTF-8 CRLF {} TypeScript Go Live

Creamos la interfaz para la realización de la tabla de mysql, define una entidad en TypeORM para representar la tabla "estudiante" en una base de datos.

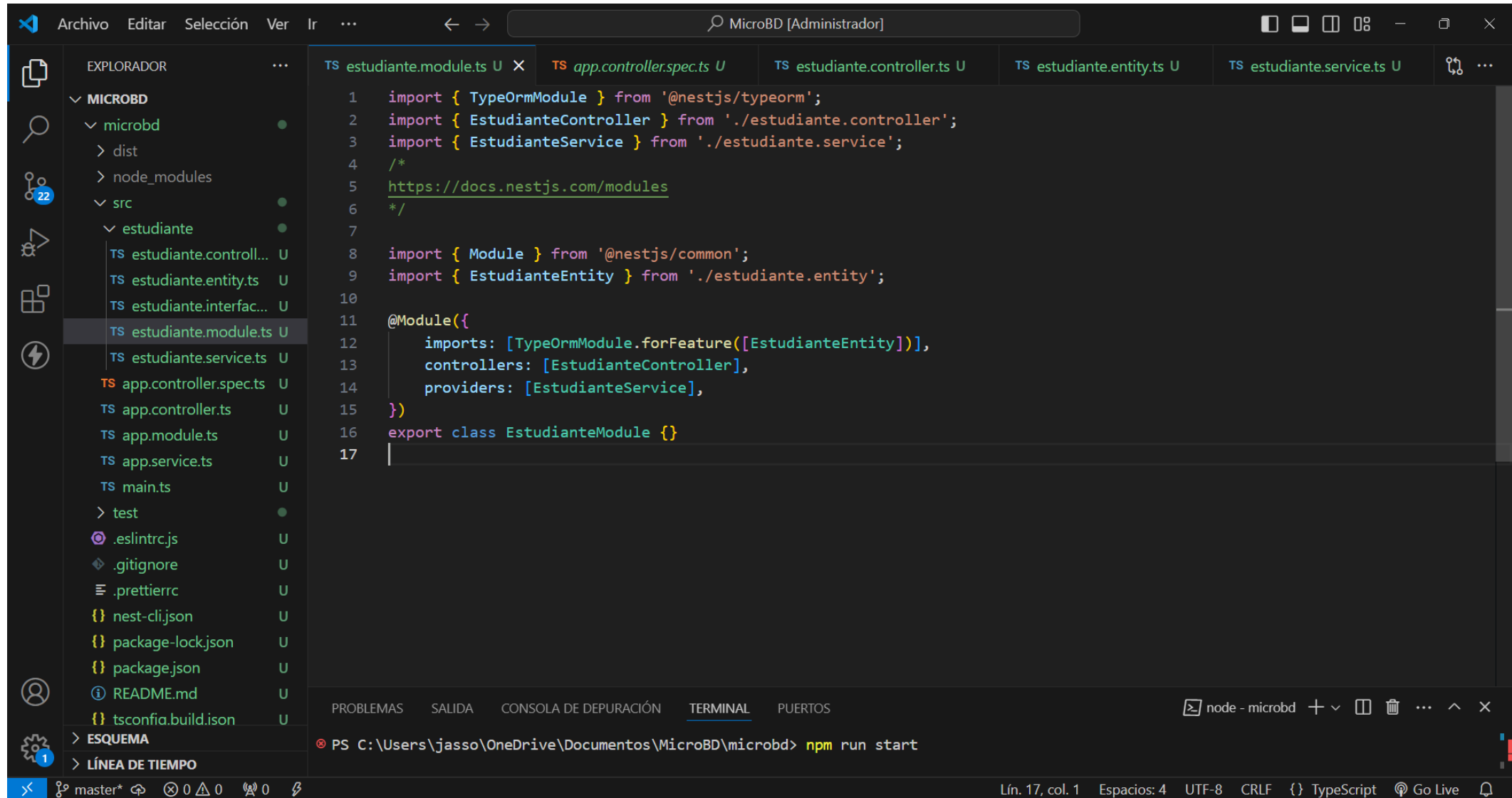
The screenshot shows the Visual Studio Code interface with the following components:

- Explorer (Left):** Displays the project structure for 'MICROBD'. The 'src' folder is expanded, showing files like 'estudiante.controller.ts', 'estudiante.entity.ts' (selected), 'estudiante.interfac...', 'estudiante.module.ts', 'estudiante.service.ts', 'app.controller.spec.ts', 'app.controller.ts', 'app.module.ts', 'app.service.ts', 'main.ts', 'test', '.eslintrc.js', '.gitignore', '.prettierrc', 'nest-cli.json', 'package-lock.json', 'package.json', 'README.md', and 'tsconfig.build.json'.
- Editor (Center):** Shows the code for 'estudiante.entity.ts'. The code defines a TypeORM entity named 'EstudianteEntity' for the 'estudiante' table. It includes imports for 'Column', 'Entity', and 'PrimaryGeneratedColumn' from 'typeorm'. The entity has five attributes: 'id' (primary key, generated), 'nombre' (string), 'apellido' (string), 'semestre' (number), and 'grupo' (string). It also has a 'fecha_registro' attribute of type 'Date' with a default timestamp value.
- Terminal (Bottom):** Shows the command 'npm run start' being executed in the 'PS C:\Users\jasso\OneDrive\Documentos\MicroBD\microbd' directory.
- Status Bar (Bottom):** Displays the current file path 'Lfn. 10, col. 14', encoding 'UTF-8', line endings 'CRLF', and the language 'TypeScript'.

Este código define una interfaz en TypeScript llamada EstudianteDto que representa el formato de los datos de un estudiante que se espera recibir en las solicitudes HTTP, especialmente en operaciones de creación y actualización.



Este código es un módulo en Nest.js que organiza y encapsula las funcionalidades relacionadas con los estudiantes, incluyendo el controlador, el servicio y la entidad, y configura la integración con TypeORM.



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

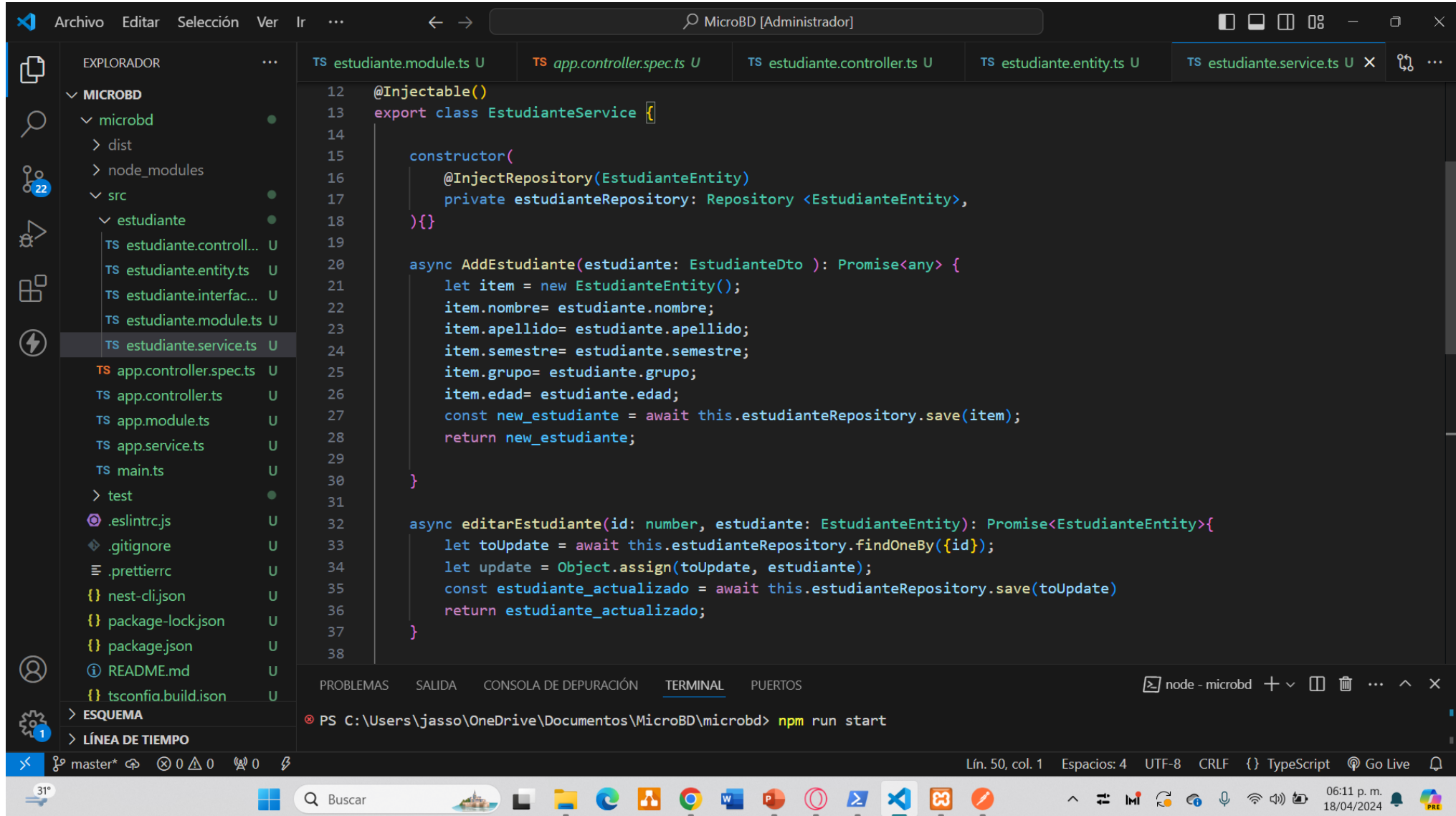
- Explorer (Left):** Shows the project structure for 'MICROBD'. The 'src' directory is expanded, showing the 'estudiante' module. The file 'TS estudiante.module.ts' is selected.
- Editor (Center):** Displays the code for 'estudiante.module.ts'. The code defines a Nest.js module that imports TypeORM, the student controller, and the student service, and registers them within the module.
- Terminal (Bottom):** Shows the command 'npm run start' being executed in the 'node - microbd' terminal.

```
1 import { TypeOrmModule } from '@nestjs/typeorm';
2 import { EstudianteController } from '../estudiante.controller';
3 import { EstudianteService } from '../estudiante.service';
4 /*
5  https://docs.nestjs.com/modules
6  */
7
8 import { Module } from '@nestjs/common';
9 import { EstudianteEntity } from '../estudiante.entity';
10
11 @Module({
12   imports: [TypeOrmModule.forFeature([EstudianteEntity])],
13   controllers: [EstudianteController],
14   providers: [EstudianteService],
15 })
16 export class EstudianteModule {}
17
```

Terminal output:

```
PS C:\Users\jasso\OneDrive\Documentos\MicroBD\microbd> npm run start
```


Este código es un servicio en Nest.js que proporciona lógica de negocio relacionada con los estudiantes, incluyendo operaciones como agregar, editar, obtener y eliminar estudiantes en la base de datos.



The screenshot displays the Visual Studio Code editor with a project named 'MicroBD [Administrador]'. The Explorer sidebar on the left shows the project structure, with the file 'TS estudiante.service.ts' selected. The main editor area shows the code for this service, which is decorated with '@Injectable()' and implements the 'EstudianteService' interface. The code includes a constructor that injects 'EstudianteRepository' and two methods: 'AddEstudiante' and 'editarEstudiante'. The 'AddEstudiante' method creates a new 'EstudianteEntity' from the provided 'EstudianteDto', sets its properties, and saves it to the repository. The 'editarEstudiante' method finds an existing entity by ID, updates its properties with the provided data, and saves the updated entity. The bottom status bar indicates the current line is 50, column 1, and the file is encoded in UTF-8 with CRLF line endings.

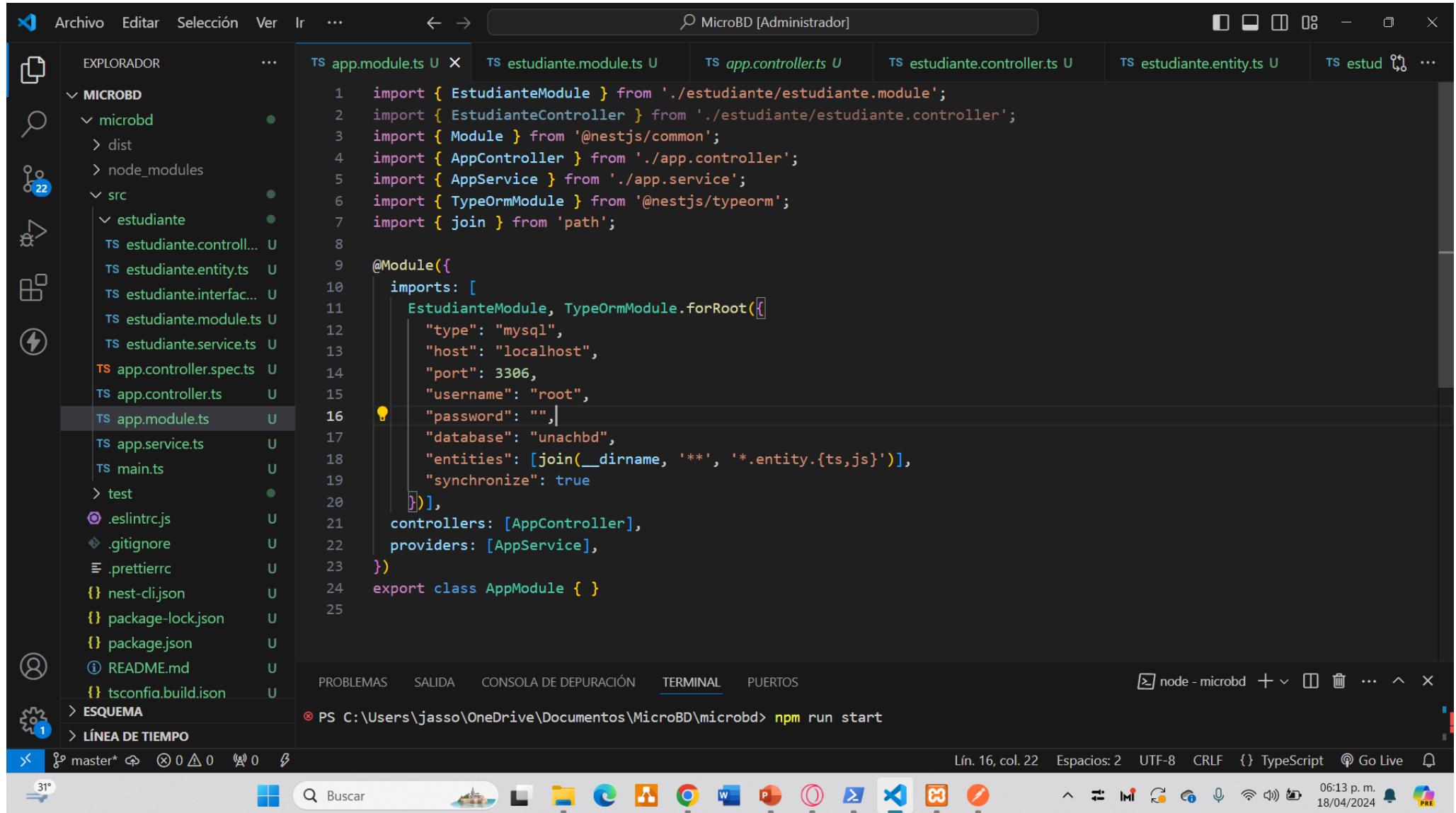
```
12 @Injectable()
13 export class EstudianteService {
14
15     constructor(
16         @InjectRepository(EstudianteEntity)
17         private estudianteRepository: Repository <EstudianteEntity>,
18     ){}
19
20     async AddEstudiante(estudiante: EstudianteDto ): Promise<any> {
21         let item = new EstudianteEntity();
22         item.nombre= estudiante.nombre;
23         item.apellido= estudiante.apellido;
24         item.semestre= estudiante.semestre;
25         item.grupo= estudiante.grupo;
26         item.edad= estudiante.edad;
27         const new_estudiante = await this.estudianteRepository.save(item);
28         return new_estudiante;
29     }
30
31     async editarEstudiante(id: number, estudiante: EstudianteEntity): Promise<EstudianteEntity>{
32         let toUpdate = await this.estudianteRepository.findOneBy({id});
33         let update = Object.assign(toUpdate, estudiante);
34         const estudiante_actualizado = await this.estudianteRepository.save(toUpdate)
35         return estudiante_actualizado;
36     }
37
38 }
```

node - microbd + - - - ^ x

PS C:\Users\jasso\OneDrive\Documentos\MicroBD\microbd> npm run start

Lín. 50, col. 1 Espacios: 4 UTF-8 CRLF {} TypeScript Go Live

Ahora en los archivos de app.module aquí este código es el módulo principal de una aplicación Nest.js que importa y configura varios módulos y servicios, incluyendo el módulo EstudianteModule que contiene la lógica relacionada con los estudiantes.



The screenshot displays the Visual Studio Code interface with a project named 'MicroBD [Administrador]'. The Explorer sidebar on the left shows the project structure:

- EXPLORADOR
 - MICROBD
 - microbd
 - dist
 - node_modules
 - src
 - estudiante
 - estudiante.controller.ts
 - estudiante.entity.ts
 - estudiante.interfac...
 - estudiante.module.ts
 - estudiante.service.ts
 - app.controller.spec.ts
 - app.controller.ts
 - app.module.ts (selected)
 - app.service.ts
 - main.ts
 - test
 - .eslintrc.js
 - .gitignore
 - .prettierrc
 - nest-cli.json
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.build.json
 - ESQUEMA
 - LÍNEA DE TIEMPO

The main editor shows the code for `app.module.ts`:

```
1 import { EstudianteModule } from './estudiante/estudiante.module';
2 import { EstudianteController } from './estudiante/estudiante.controller';
3 import { Module } from '@nestjs/common';
4 import { AppController } from './app.controller';
5 import { AppService } from './app.service';
6 import { TypeOrmModule } from '@nestjs/typeorm';
7 import { join } from 'path';
8
9 @Module({
10   imports: [
11     EstudianteModule, TypeOrmModule.forRoot({
12       "type": "mysql",
13       "host": "localhost",
14       "port": 3306,
15       "username": "root",
16       "password": "",
17       "database": "unachbd",
18       "entities": [join(__dirname, '**', '*.entity.{ts,js}')],
19       "synchronize": true
20     }
21   ],
22   controllers: [AppController],
23   providers: [AppService],
24 })
25 export class AppModule { }
```

The bottom status bar indicates the current position: 'Lín. 16, col. 22'. The terminal at the bottom shows the command: `PS C:\Users\jasso\OneDrive\Documentos\MicroBD\microbd> npm run start`.

```
9  @Module({
10    imports: [
11      EstudianteModule, TypeOrmModule.forRoot({
12        "type": "mysql",
13        "host": "localhost",
14        "port": 3306,
15        "username": "root",
16        "password": "",
17        "database": "unachbd",
18        "entities": [join(__dirname, '**', '*.entity.{ts,js}')],
19        "synchronize": true
20      }]),
21    controllers: [AppController],
22    providers: [AppService],
23  })
24  export class AppModule { }
25
```

Aquí ponemos en nombre de nuestra base donde se van a crear las tablas.

Ejecutamos el programa para que funcione postman

```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS  node - microbd + ▾ □ 🗑️ ⋮ ^ ✕

PS C:\Users\jasso\OneDrive\Documentos\MicroBD\microbd> npm run start

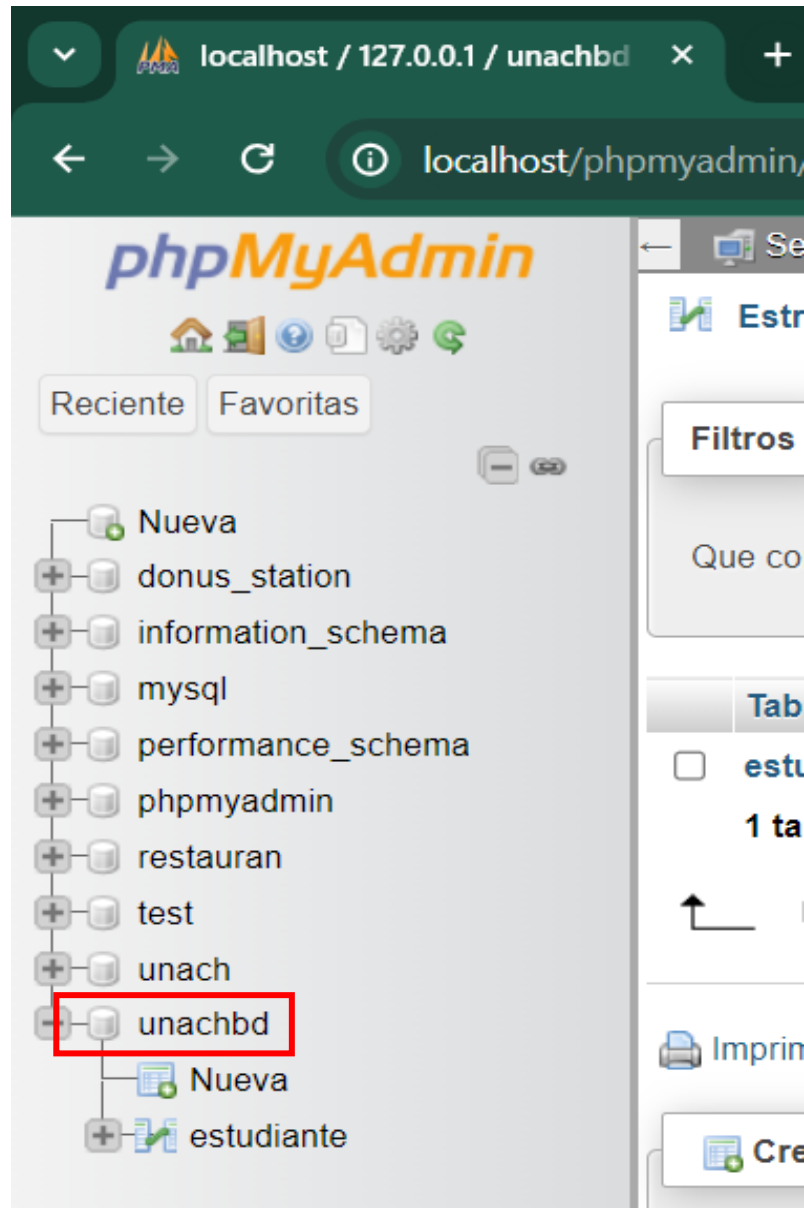
> microbd@0.0.1 start
> nest start

[Nest] 20752 - 18/04/2024, 5:12:00 p.m. LOG [NestFactory] Starting Nest application...
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [InstanceLoader] TypeOrmModule dependencies initialized +194ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [InstanceLoader] AppModule dependencies initialized +0ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [InstanceLoader] TypeOrmCoreModule dependencies initialized +404ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [InstanceLoader] TypeOrmModule dependencies initialized +1ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [InstanceLoader] EstudianteModule dependencies initialized +0ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [RoutesResolver] AppController {/}: +8ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [RouterExplorer] Mapped {/, GET} route +6ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [RoutesResolver] EstudianteController {/estudiantes}: +0ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [RouterExplorer] Mapped {/estudiantes, GET} route +1ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [RouterExplorer] Mapped {/estudiantes, POST} route +1ms
[Nest] 20752 - 18/04/2024, 5:12:01 p.m. LOG [RouterExplorer] Mapped {/estudiantes/:id, PUT} route +0ms

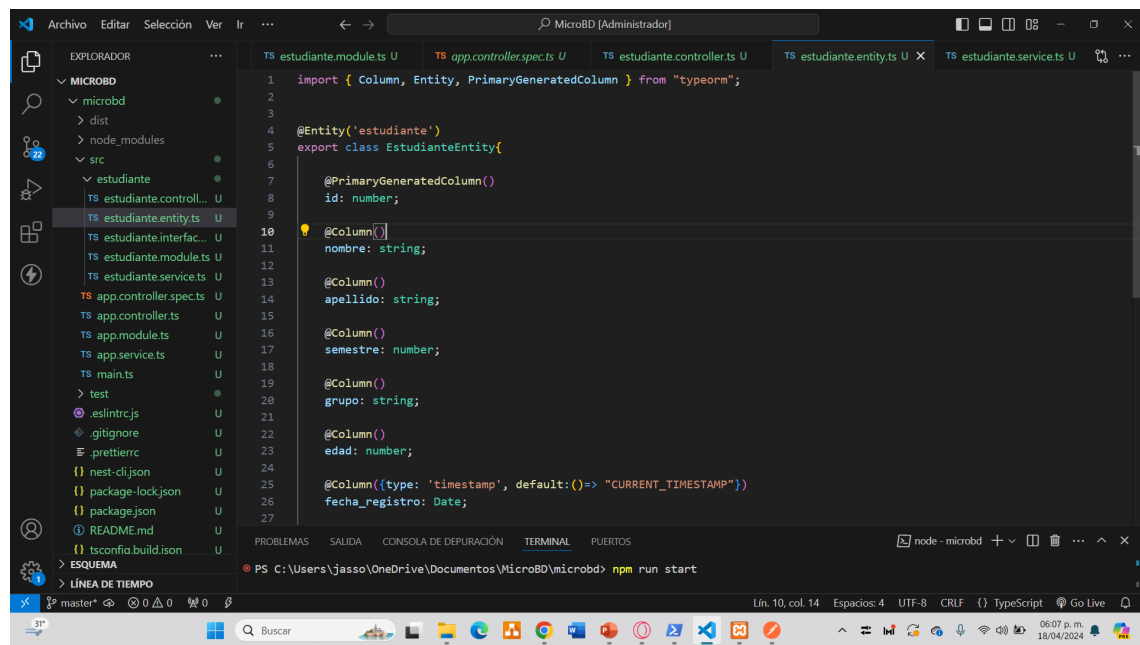
Lín. 1, col. 1  Espacios: 2  UTF-8  CRLF  {} TypeScript  📶 Go Live  🔔
```

PASAMOS A PRUEBA A LA PRUEBA

Creamos solo la base de
datos



Como se creo las columnas y el tipo de dato que es cuando se hace la conexión automáticamente se hace las columnas en la base



The screenshot shows a VS Code editor with a TypeScript file named 'estudiante.entity.ts'. The code defines an entity class 'EstudianteEntity' with the following properties:

```
import { Column, Entity, PrimaryGeneratedColumn } from "typeorm";

@Entity('estudiante')
export class EstudianteEntity {
  @PrimaryGeneratedColumn()
  id: number;

  @Column()
  nombre: string;

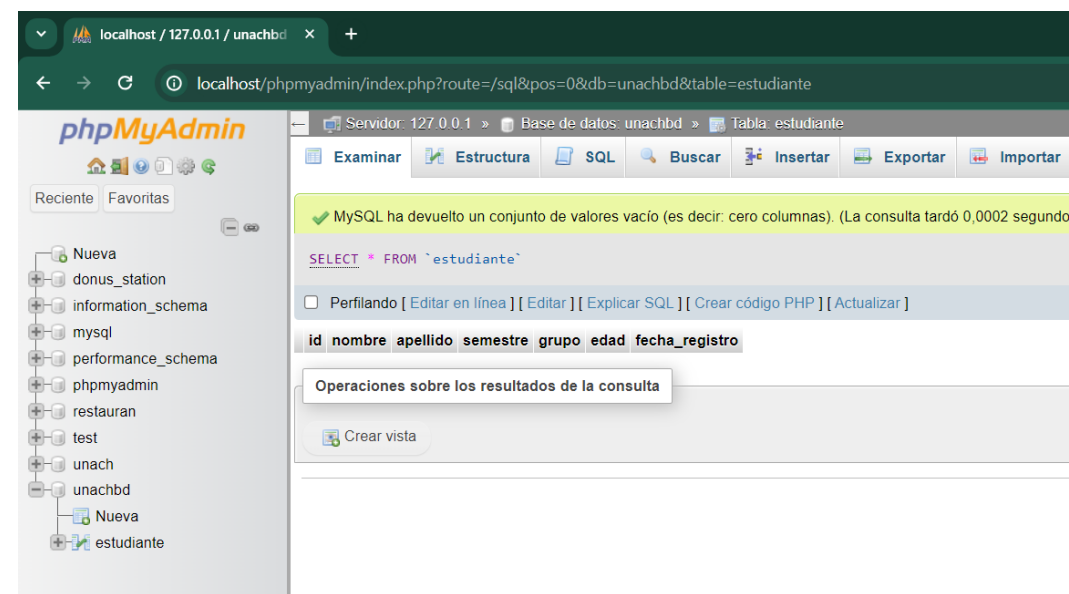
  @Column()
  apellido: string;

  @Column()
  semestre: number;

  @Column()
  grupo: string;

  @Column()
  edad: number;

  @Column({type: 'timestamp', default: () => "CURRENT_TIMESTAMP"})
  fecha_registro: Date;
}
```



AHORA EN POSTMAN

GET Obtener Estudiantes

microBD / Obtener Estudiantes

GET http://localhost:3000/estudiantes

Send

microBD / Obtener Estudiantes

GET http://localhost:3000/estudiantes

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Status: 200 OK Time: 69 ms Size: 762 B Save as example

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "nombre": "Jasson Jare",
5     "apellido": "clemente",
6     "semestre": 6,
7     "grupo": "m",
8     "edad": 19,
9     "fecha_registro": "2024-04-18T22:30:38.000Z"
10  },
11  {
12    "id": 2,
13    "nombre": "Miguel",
14    "apellido": "Corzo",
15    "semestre": 6,
16    "grupo": "m",
17    "edad": 20,
18    "fecha_registro": "2024-04-18T22:35:43.000Z"
19  }
20 ]
```



mysql

performance_schema phpmyadmin restauran test unach unachbd Nueva estudiante

Mostrar todo Número de filas: 25 Filtrar filas: Buscar en esta tabla Ordenar según la clave: Ninguna

Opciones extra

	id	nombre	apellido	semestre	grupo	edad	fecha_registro
<input type="checkbox"/>	1	Jasson Jare	clemente	6	m	19	2024-04-18 16:30:38
<input type="checkbox"/>	2	Miguel	Corzo	6	m	20	2024-04-18 16:35:43
<input type="checkbox"/>	4	Brayan	Alegria de la cruz	6	m	23	2024-04-18 17:20:21
<input type="checkbox"/>	6	Iker	Fernandez	6	m	25	2024-04-18 17:50:38

Seleccionar todo Para los elementos que están marcados: Editar Copiar Borrar Exportar

POST Agregar Nuevos Estudiantes

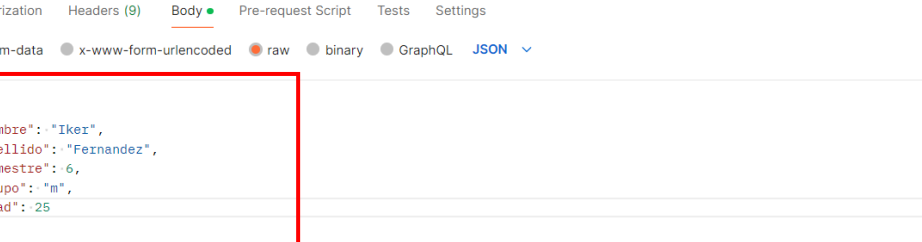
POST

▼

http://localhost:3000/estudiantes

Send

▼



The screenshot displays a REST client interface with the following details:

- Request:** Method `POST`, URL `http://localhost:3000/estudiantes`.
- Response:** Status `201 Created`, Time `158 ms`, Size `367 B`.
- Response Body (JSON):**

```
{
  "nombre": "Iker",
  "apellido": "Fernandez",
  "semestre": 6,
  "grupo": "m",
  "edad": 25,
  "id": 6,
  "fecha_registro": "2024-04-18T23:50:38.000Z"
}
```

☐ Mostrar todo | Número de filas: 25 ▼ | Filtrar filas: | Ordenar según la clave: Ninguna

Opciones extra

				id	nombre	apellido	semestre	grupo	edad	fecha_registro
<input type="checkbox"/>				1	Jasson Jare	clemente	6 m		19	2024-04-18 16:30:38
<input type="checkbox"/>				2	Miguel	Corzo	6 m		20	2024-04-18 16:35:43
<input type="checkbox"/>				4	Brayan	Alegria de la cruz	6 m		23	2024-04-18 17:20:21
<input checked="" type="checkbox"/>				6	Iker	Fernandez	6 m		25	2024-04-18 17:50:38

☐ Seleccionar todo | Para los elementos que están marcados: Editar | Copiar | Borrar | Exportar

(EL id y fecha_registro no se pone en el doby ya que eso se genera automáticamente)

DEL Eliminar Estudiante con Id

DELETE

http://localhost:3000/estudiantes/6

Send

DELETE http://localhost:3000/estudiantes/6

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 24 ms Size: 145 B Save as example

Pretty Raw Preview Visualize Text

1

				id	nombre	apellido	semestre	grupo	edad	fecha_registro
<input type="checkbox"/>	Editar	Copiar	Borrar	1	Jasson	Jare clemente	6	m	19	2024-04-18 16:30:38
<input type="checkbox"/>	Editar	Copiar	Borrar	2	Miguel	Corzo	6	m	20	2024-04-18 16:35:43
<input type="checkbox"/>	Editar	Copiar	Borrar	4	Brayan	Alegria de la cruz	6	m	23	2024-04-18 17:20:21
<input type="checkbox"/>	Editar	Copiar	Borrar	6	Iker	Fernandez	6	m	25	2024-04-18 17:50:38

				id	nombre	apellido	semestre	grupo	edad	fecha_registro
<input type="checkbox"/>	Editar	Copiar	Borrar	1	Jasson	Jare clemente	6	m	19	2024-04-18 16:30:38
<input type="checkbox"/>	Editar	Copiar	Borrar	2	Miguel	Corzo	6	m	20	2024-04-18 16:35:43
<input type="checkbox"/>	Editar	Copiar	Borrar	4	Brayan	Alegria de la cruz	6	m	23	2024-04-18 17:20:21

Solo con el id se borra el registro de la tabla

PUT Editar datos con id

PUT ⌵ http://localhost:3000/estudiantes/4 Send ⌵

Con el id y el body se actualiza el registro

PUT ⌵ http://localhost:3000/estudiantes/4

Params Authorization Headers (9) **Body** • Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ⌵

```
1 {
2   ...
3   "nombre": "Brayan Fermin",
4   "apellido": "Alegria de la cruz",
5   "semestre": 6,
6   "grupo": "m",
7   "edad": 30
8 }
9
10 }
```



PUT ⌵ http://localhost:3000/estudiantes/4

Params Authorization Headers (9) **Body** • Pre-request Script Tests Settings










☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ⌵

```
1 {
2   ...
3   "nombre": "Brayan Fermin",
4   "apellido": "Alegria de la cruz",
5   "semestre": 6,
6   "grupo": "m",
7   "edad": 30
8 }
9 }
```








Body Cookies Headers (7) Test Results ⌵ Status: 200 OK Time: 39 ms Size: 381 B

Pretty Raw Preview Visualize **JSON** ⌵

```
1 {
2   "id": 4,
3   "nombre": "Brayan Fermin",
4   "apellido": "Alegria de la cruz",
5   "semestre": 6,
6   "grupo": "m",
7   "edad": 30,
8   "fecha_registro": "2024-04-18T23:20:21.000Z"
9 }
```

↩️ ⬅️ ➡️ ⏪ ⏩ ⏴ ⏵			id	nombre	apellido	semestre	grupo	edad	fecha_registro
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	1	Jasson Jare	clemente	6 m	19	2024-04-18 16:30:38
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	2	Miguel	Corzo	6 m	20	2024-04-18 16:35:43
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	4	Brayan	Alegria de la cruz	6 m	23	2024-04-18 17:20:21



← T →												id	nombre	apellido	semestre	grupo	edad	fecha_registro
<input type="checkbox"/>		Editar		Copiar		Borrar	1	Jasson Jare	clemente	6	m	19	2024-04-18 16:30:38					
<input type="checkbox"/>		Editar		Copiar		Borrar	2	Miguel	Corzo	6	m	20	2024-04-18 16:35:43					
<input type="checkbox"/>		Editar		Copiar		Borrar	4	Brayan Fermin	Alegria de la cruz	6	m	30	2024-04-18 17:20:21					