

```
C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [Versión 10.0.22621.1848]
(c) Microsoft Corporation. Todos los derechos reservados.

E:\back-end>tsc -v
Version 5.0.4

E:\back-end>npm init -y
Wrote to E:\back-end\package.json:

{
  "name": "back-end",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

E:\back-end>code .
E:\back-end>tsc --init
Created a new tsconfig.json with:
```

```
"author": "",
"license": "ISC"
}

E:\back-end>code .
E:\back-end>tsc --init
Created a new tsconfig.json with:

target: es2016
module: commonjs
strict: true
esModuleInterop: true
skipLibCheck: true
forceConsistentCasingInFileNames: true

You can learn more at https://aka.ms/tsconfig

E:\back-end>npm install -g promise-mysql express nodemon morgan cors crypto-js jsonwebtoken validator
added 99 packages, and changed 32 packages in 19s

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

E:\back-end>
```

Alumno: Juan Luis Negrete Labrada

```
C:\WINDOWS\system32\cmd. x + v
Created a new tsconfig.json with:
  target: es2016
  module: commonjs
  strict: true
  esModuleInterop: true
  skipLibCheck: true
  forceConsistentCasingInFileNames: true

You can learn more at https://aka.ms/tsconfig

E:\back-end>npm install -g promise-mysql express nodemon morgan cors crypto-js jsonwebtoken validator
added 99 packages, and changed 32 packages in 19s

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

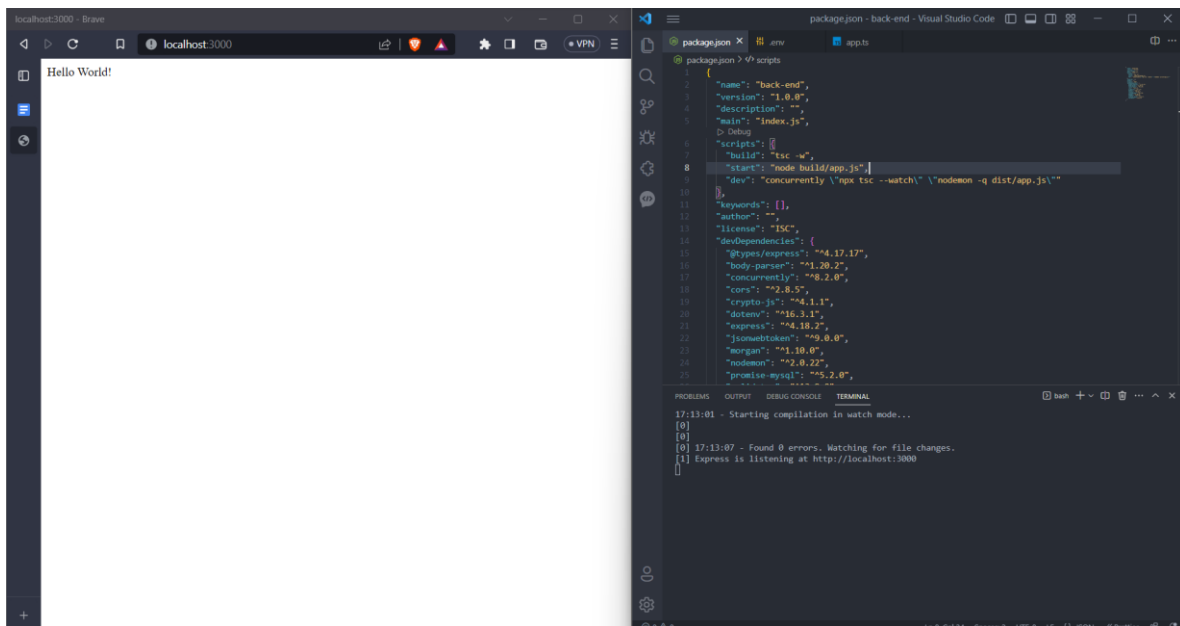
E:\back-end>npm i -g promise-mysql
changed 16 packages in 5s

E:\back-end>npm i promise-mysql
added 16 packages, and audited 17 packages in 2s

found 0 vulnerabilities

E:\back-end>
```

```
Debug
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "build": "tsc -w",
  "start": "node build/app.js"
},
```



Alumno: Juan Luis Negrete Labrada

Comprobar la ruta de la aplicación

The screenshot shows a REST client interface with a dark theme. At the top, a tab displays 'GET http://localhost:3000'. Below this, the URL 'http://localhost:3000' is entered. The interface has several tabs: 'Params', 'Authorization', 'Headers (6)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Params' tab is active, showing a section for 'Query Params' with a table that has two columns: 'Key' and 'Value'. The table is currently empty. Below the 'Params' tab, there are more tabs: 'Body', 'Cookies', 'Headers (8)', and 'Test Results'. The 'Body' tab is active, showing a list of items. The first item is '1 Invocando autenticación'. At the bottom of the 'Body' tab, there are buttons for 'Pretty', 'Raw', 'Preview', 'Visualize', and a dropdown menu for 'HTML'.

GET http://localhost:3000

http://localhost:3000

GET http://localhost:3000

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

Key	Value
Key	Value

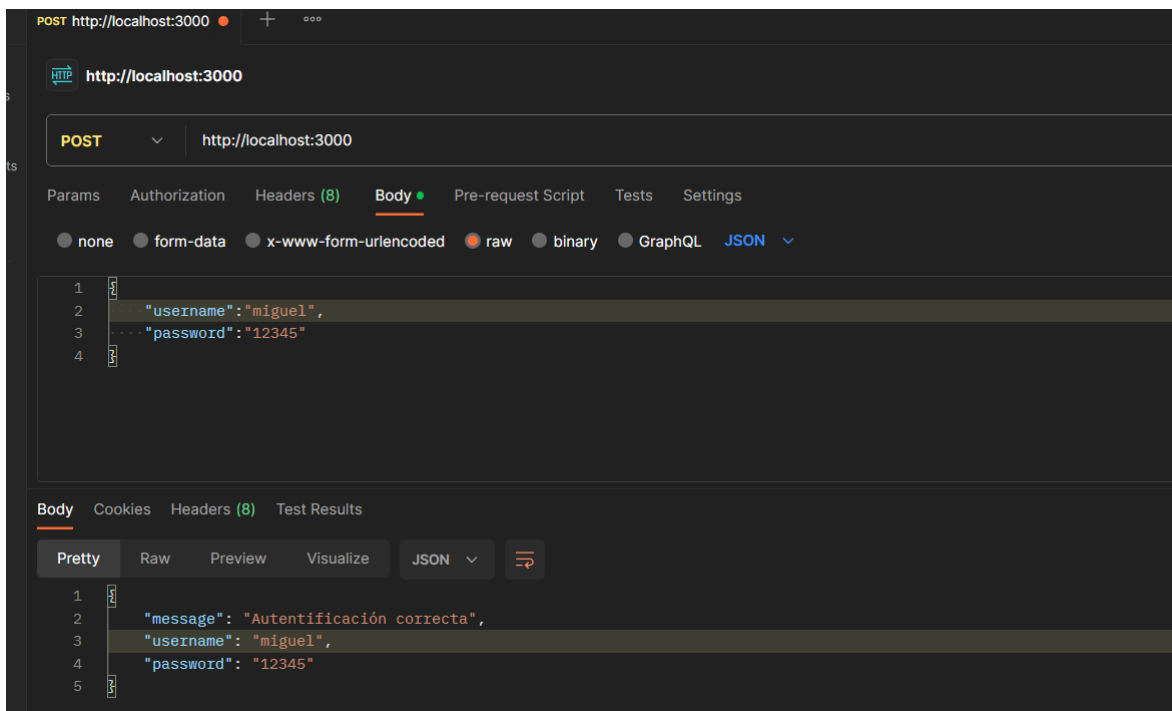
Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize HTML

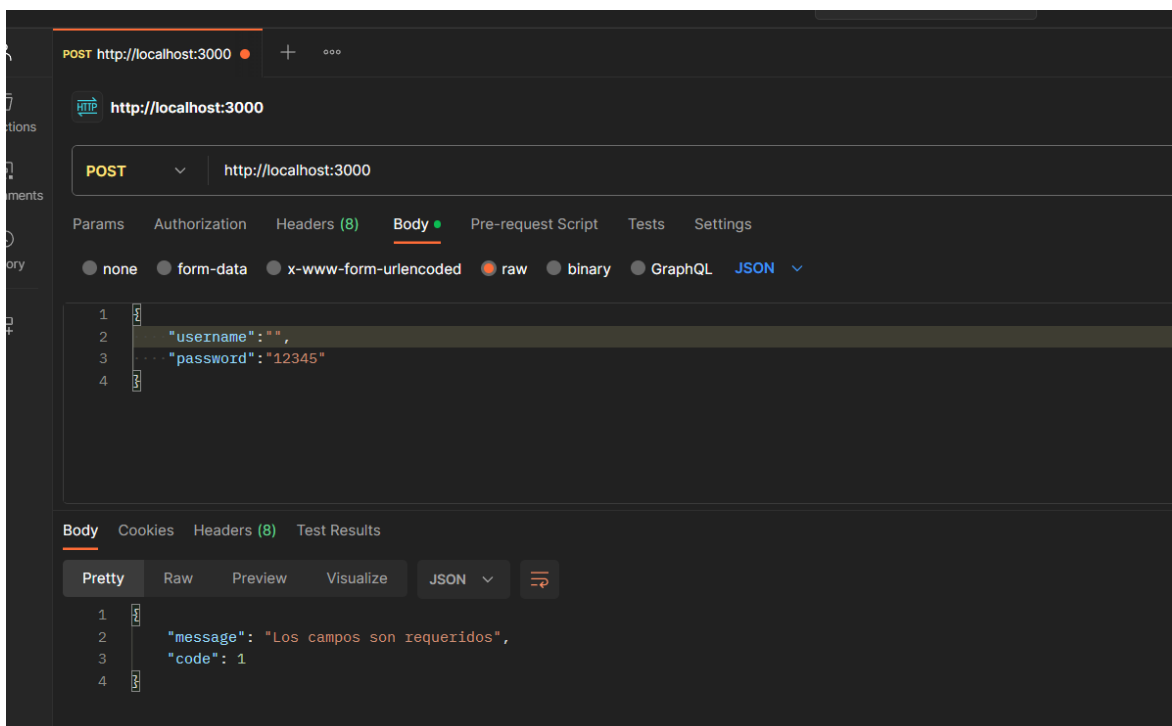
1 Invocando autenticación

Alumno: Juan Luis Negrete Labrada

Inicio de sesión



Validación de campos



Alumno: Juan Luis Negrete Labrada

Inserción de nuevo usuario

HTTP New Collection / http://localhost:3000/user/insert

POST http://localhost:3000/user/insert

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary ● GraphQL **JSON** ✓

```
1 {
2   ... "username": "AlanMozo",
3   ... "password": "123456",
4   ... "role": "administrador"
5 }
6
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ✓

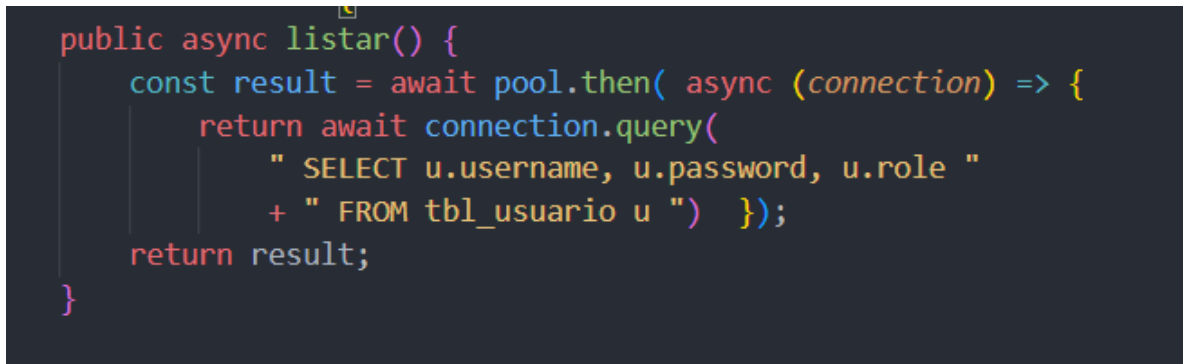
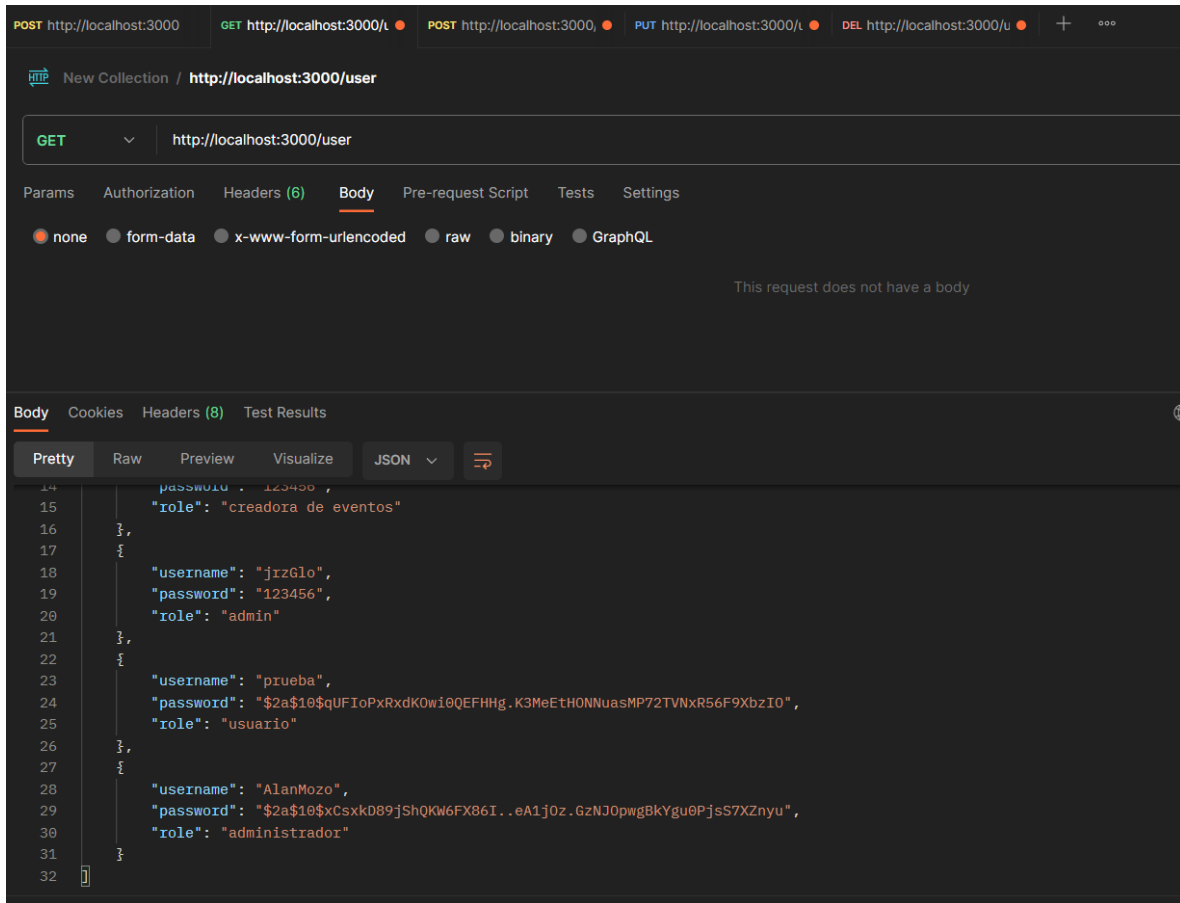
```
1 {
2   "message": "Los datos se guardaron correctamente",
3   "code": 0
4 }
```

```
public async insertar(usuario: any) {
  const result = await pool.then( async (connection) => {
    return await connection.query(
      " INSERT INTO tbl_usuario SET ? ", [usuario]
    );
  });
  return result;
}
```

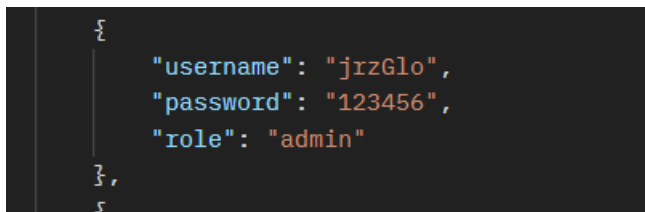
```
this.router.post('/insert', usuarioController.insertar);
```

Alumno: Juan Luis Negrete Labrada

Listar los registros de los datos de la base



Actualizar registro



The screenshot displays a REST client interface with a dark theme. At the top, it shows 'New Collection / http://localhost:3000/user/update'. Below this, the request method is set to 'PUT' and the URL is 'http://localhost:3000/user/update/jrzGlo'. The 'Body' tab is selected, showing a JSON payload:

```
{  "username": "jrzGlo",  "password": "019690"}
```

. The response section below shows the 'Body' tab with a 'Pretty' view of the response:

```
{  "message": "Los datos se actualizaron correctamente",  "code": 0}
```

. A second screenshot at the bottom shows a JSON object:

```
{  "username": "jrzGlo",  "password": "$2a$10$uKxz10vetTLnmcHuFTqEl.1b69QViZzb4Df9T6.k/mLYqB6pITZ4u",  "role": "admin"}
```

HTTP New Collection / http://localhost:3000/user/update

PUT http://localhost:3000/user/update/jrzGlo

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "username": "jrzGlo",
3   ... "password": "019690"
4   ...
5 }
6
7
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Los datos se actualizaron correctamente",
3   "code": 0
4 }
```

```
{
  "username": "jrzGlo",
  "password": "$2a$10$uKxz10vetTLnmcHuFTqEl.1b69QViZzb4Df9T6.k/mLYqB6pITZ4u",
  "role": "admin"
},
```

Alumno: Juan Luis Negrete Labrada

```
public async actualizar(usuario: any, username: string) {  
    const result = await pool.then( async (connection) => {  
        return await connection.query(  
            "UPDATE tbl_usuario SET ? WHERE username = ?",  
            [usuario, username]  
        );  
    });  
    return result;  
}
```

```
this.router.put('/update/:username',usuarioController.actualizar);
```

Eliminación de usuario

The screenshot shows a REST client interface with a new collection named 'http://localhost:3000/user/delete'. A DELETE request is configured for the URL 'http://localhost:3000/user/delete/prueba'. The 'Headers' tab is active, showing 8 hidden headers. The 'Body' tab is also active, displaying a JSON response in 'Pretty' format:

```
{  
  "message": "Los datos se eliminaron correctamente",  
  "code": 0  
}
```



```
public async eliminar(username: string) {  
  console.log('Eliminando DAO');  
  const result = await pool.then( async (connection) => {  
    return await connection.query(  
      'DELETE FROM tbl_usuario WHERE username = ?', [username]  
    );  
  });  
  return result;  
}
```

```
this.router.delete('/delete/:username',usuarioController.eliminar);
```